MASTER EN SCIENCES INFORMATIQUES À FINALITÉ SPÉCIALISÉE EN SOFTWARE ENGINEERING

Prédiction de la demande en électricité via l'utilisation du machine learning

Pierre, Baptiste

Award date:
2020

Awarding institution:
Universite de Namur

UNIVERSITÉ DE NAMUR

FACULTÉ D'INFORMATIQUE

Prédiction de la demande en électricité via l'utilisation du machine learning

Pierre Baptiste

RUE GRANDGAGNAGE, 21 ● B-5000 NAMUR(BELGIUM)

Remerciements

Ce mémoire ayant eu lieu dans des conditions particulièrement difficiles je tiens à remercier avant tout Mr Vanhoof qui a fait tout ce qui était en son pouvoir pour me permettre de continuer mon mémoire qui avait pris un très mauvais tournant avec l'annonce du confinement. Mr Frenay qui malgré son emploi du temps fort chargé a su trouver le temps pour me guider et me corriger sans quoi je n'aurai probablement pas pu rendre un mémoire correcte dans les temps. Mr Divina de l'université Pablo Olavide qui m'a aidé à régler pas mal de soucis que ce soit au niveau de mon mémoire comme de mon erasmus en général. Et pour finir Mr Álvarez ainsi que toute l'équipe du laboratoire de Data science de l'upo de m'avoir si bien accueilli et avec qui je regrette seulement de ne pas avoir pu travailler plus longtemps.

Résumé

Ce mémoire traite de la question de la prédiction de la consommation électrique d'un quartier résidentiel à l'aide du machine learning et plus particulièrement ici des réseaux de neurones artificiel et des random forest. Ce document contient un état de l'art du sujet ainsi qu'un résumé de la problématique dont il est question, une analyse de la performance des modèles en terme de prédiction et comment optimiser ceux-ci pour obtenir les résultats les plus précis possibles.Le tout en influent sur des paramètres tels que la période d'observation donnée au modèle, le temps entre chaque observation ou encore la taille des séries temporelles passées au modèle durant la phase d'entraînement.

Mots-clefs

prédiction, time-series, réseau de neurones, machine learning, électricité, ARMA, ARIMA, random forest

Table des matières

1. Introduction

Avec l'augmentation de la population mondiale, la montée de l'utilisation des énergies vertes et notamment via le biais des véhicules électriques, la question de la gestion de cette demande croissante en énergie fait partie des préoccupations centrales pour les années à venir [POP08]. À tout cela vient s'ajouter la volonté des citoyens de mieux gérer leurs consommations personnelles que ce soit pour des raisons écologiques ou économiques.
De cette demande de contrôle et de besoin d'information (que ce soit du côté du fournisseur ou du client) est ressorti la notion de "smart city" [KMM14]. Une smart city désigne une zone urbaine utilisant l'internet des objets afin de récolter des données qui seront ensuite utilisées à des fins d'efficacité (comme dans le cadre de ce mémoire) ou de contrôle dans le cadre des habitants cherchant à mieux réguler leurs consommations. L'idée principale réside dans la récolte et le partage d'un maximum de données principalement du consommateur vers le fournisseur, mais aussi inversement.
Afin de réaliser cela dans le cadre de la gestion de l'énergie, la composante principale est le "smart meter" [WR96]. Un smart meter et un compteur électrique qui en plus de relever la consommation électrique va aussi transmettre des informations sur celle-ci en temps réel au fournisseur. L'ensemble de ces compteurs intelligents mêlés à d'autres outils de récolte et de partage d'information s'appelle "smart grid" [Far10]. Les smart grids permettent de passer d'un système de distribution d'énergie simple à un système où il y a un retour d'information du client au fournisseur permettant de rendre la distribution plus efficace, fiable et sécurisée. Dans ce mémoire le but concerne l'utilisation de ces données récoltées par le biais du machine learning afin de pouvoir prédire le trafique électrique en vue d'éviter une surcharge de celui-ci.
Ce travail a pour but de répondre à la problématique de My Electric Avenue [Ave] qui est un projet ayant pour objectif d'évaluer l'impact de la croissance de l'utilisation des véhicules électriques sur la demande globale en électricité. La croissance de l'utilisation des véhicules électriques implique une croissance simultanée de la demande en électricité et pourrait à terme poser un problème de surcharge des câbles n'étant plus assez puissant que pour fournir la quantité suffisante d'énergie demandée. Pour

régler ce problème de surcharge My electric avenue cherche à proposer une solution influant sur la charge des véhicules électriques lorsque la demande excède la capacité du réseau. Afin d'aider dans la recherche de ce genre de solution la société a donc mis à disposition des données sur la consommation électrique de plusieurs populations avec l'aide d'appareils mesurant celle-ci que l'on peut qualifier de smart meters. Le but de ce mémoire est de prédire le plus efficacement possible à court terme (soit 2 et 20 heures à l'avance) la demande en énergie du réseau afin que lorsque celle-ci dépasse un certain seuil des solutions de gestions efficaces comme par exemple la mise en veille du chargement des véhicules électriques jusqu'à ce que la consommation repasse en dessous d'un autre seuil soient appliquées pour éviter la surcharge du réseau électrique. Ce sujet ayant déjà été couvert par de nombreux travaux l'intérêt de celui-ci va être d'appliquer ceux ayant produits les meilleurs résultats de manière constante à travers le temps de part leurs fonctionnements.

Afin de produire les meilleurs résultats la première étape est de consulter la littérature afin de trouver les méthodes ayant déjà par le passé produit des résultats intéressants. L'intuition poursuivie dans ce travail s'est donc portée sur les modèles "AutoRegressive Moving Average" (ARMA) [Whi51] et ses dérivés ARIMA et SARIMA qu'on explicitera par la suite, la "Random Forest" (RF) [Tin95] et les réseaux de neurones artificiels (ANN) [War43] dont plus particulièrement ceux de type "Long short term memory" LSTM [Gre+17].

2. État de l'art

Dans un travail de référencement Fikirte Zemene et Vijayshri Khedkar [ZK17] ont recensé une multitude de travaux ayant été réalisés à l'aide de données produites par des smart meters et ont souligné les limites ainsi que les points forts de chacune des études recensées. La gestion de l'énergie est une des caractéristiques les plus flexibles comme vitales de la société actuelle ce qui place celle-ci comme le sujet le plus prometteur en terme d'amélioration pour les années à venir. L'utilisation du machine learning dans le cadre des smart grids permettrait l'application de systèmes comme celui que My electric avenue développe [Ave] afin de résoudre les problèmes se posant lors de la croissance de la demande.

Damien Zufferey et Christophe Gisler ainsi que Omar Abou Khaled et Jean Hennebert ont réalisé un travail permettant à travers le machine learning de reconnaître le type d'appareils utilisés selon leurs consommations afin de donner un feedback aux utilisateurs sur ceux-ci [Zuf+12]. Par exemple pour permettre un nouvel arrangement plus économique, certains appareils ne nécessitant pas d'être mis en charge en journée pourraient être branchés uniquement durant la nuit où les tarifs sont plus bas. Pour reconnaître ces différents appareils, des patterns de comportement ont été établis en observant la consommation individuelle de 5 catégories d'appareils et en essayant d'associer la consommation globale à la combinaison la plus proche de ces patterns individuels. Ce travail comporte un intérêt sur le long terme en relation avec ce mémoire car tout comme les appareils visés par les recherches citées précédemment les véhicules électriques sont concernés. Comme My electric avenue l'a décrit dans l'énoncé de son projet la possession d'un véhicule électrique impacte de plus en plus la demande globale d'électricité et le fait de pouvoir en tant que fournisseur identifier les quartiers possédant un grand nombre de ces véhicules serait un paramètre que l'on pourrait ajouter au modèle dans un futur travaille sur ce sujet.

Concernant maintenant les algorithmes de machine learning pouvant être utilisés à des fins d'amélioration de la gestion de la consommation électrique plusieurs travaux de comparaisons ont été réalisés et appliqués à des cas différents. Ce mémoire explore plusieurs techniques de machine learning afin de pouvoir proposer une comparaison complète des différents

avantages et désavantages que proposent les approches explorées pour prédire la consommation des données produites par My Electric avenue. Le but de cet état de l'art est d'une part de présenter des travaux variés en terme d'approche du problème de la prédiction de données telles que la demande en énergie électrique, mais aussi de parcourir une certaine ligne du temps des travaux réalisés sur le sujet et les techniques utilisées.
En 2015 le groupe de recherche en Soft Computing de l'université technique de Catalogne a réalisé un travail de comparaison de différentes approches de Machine Learning utilisées à différents niveaux de consommation [Jur+15] afin d'essayer d'en déduire lesquels seraient les moins coûteuses pour le meilleur degré de fiabilité possible. Pour ce faire 3 bâtiments avec des profils de consommation différents ont été étudiés durant une période d'un an et les modèles : RF (Random Forest), ANN (Artificial Neural Network), FIR (Fuzzy Inductive Reasonning) et ARIMA (AutoRegressive Integrated Moving Average) ont été appliqués aux données. De cela est ressorti que le FIR de manière générale était le plus précis et le plus rapide pour prédire la consommation individuelle d'un bâtiment suivi par RF et ANN que l'on étudiera dans ce travail. Selon les auteurs cela est dû au fait que cette méthodologie contrairement aux autres ne nécessite pas de paramétrage ce qui la rend plus rapide mais aussi dû à sa nature propre à synthétiser les données empiriques et à les garder sous forme de patterns contrairement aux autres qui sont basés plus sur l'entraînement. Un autre résultat intéressant est que ARIMA ne produit pas de grandes erreurs en général mais sa précision reste faible c'est-à-dire que ses prédictions approximeront le comportement des données mais sans pour autant être très fidèles (comme une droite approximerai le comportement d'un nuage de points dans un graphique). Dans le cas de ce projet cela en fait la parfaite référence afin de mesurer l'efficacité des autres méthodes étudiées.
En 2016 M.A. Rafe Biswas, Melvin D. Robinson et Nelson Fumo ont eux réalisé un travail sur l'utilisation de réseaux de neurones pour prédire la consommation de bâtiments résidentiels [BRF16]. Les données utilisées ont été récoltées en observant une maison non-habitée appartenant à l'université TxAIRE et ce durant une période de 3 mois. Le fait que cette maison n'était pas habitée joue un rôle important dans les données car le facteur humain n'était pas pris en compte. Dans le cadre de ce travail les réseaux de neurones utilisés étaient du type "feed-forward" et ne travaillaient pas avec des séries temporelles comme dans ce cas d'étude. Une série temporelle consiste en une suite de valeurs qu'a/ont pris la/les variable(s) observée(s) à différents moment dans le temps et de préférence à intervalle régulier comme c'est le cas dans ce mémoire (ici la variable observée est la demande globale en électricité). Il s'agît de l'évolution au cours du temps d'une ou plusieurs variables. Il est donc intéressant de souligner la conclusion de cet article qui est que les réseaux de neurones donnent de bons résultats car ceux-ci se prêtent bien aux problèmes

non-linéaires tel que la consommation électrique d'un bâtiment résidentiel. En Juin 2017 Riccardo Bonetto et Michele Rossi ont à leurs tour menés un travail de recherche sur les différents algorithmes disponibles et leurs avantages et désavantages concernant la prédiction de la consommation électrique d'un ménage [BR17]. Dans ce travail les modèles mis à l'oeuvre étaient : SVM (Support Vector Machine), NAR (Non AutoRegressive Neural Network), LSTM (Long Short Term Memory Neurals Network) et ARMA (que l'on utilise dans ce mémoire) qui a servi ici de référence pour la comparaison. Ce mémoire contenant une partie utilisant le LSTM il est intéressant de souligner que l'algorithme appliqué avec celui-ci était l'ADAGRAD ou algorithme du gradient stochastique ce qui n'est pas celui utilisé dans le cadre de ce mémoire. Les résultats des recherches cependant ont montrés que tous ces algorithmes surpassaient celui de L'ARMA mais sans réellement que l'un d'entre eux se démarque pour autant.

En Septembre 2017 une équipe composée de chercheurs provenant de plusieurs instituts chinois ainsi que de Suède ont collaboré afin de mener un travail d'une assez grande ampleur consistant en l'analyse d'une multitude de techniques de prédiction et de classification dite "data-driven" afin de pouvoir comparer ces dernières [Wei+18]. Beaucoup de méthodes ont été analysées dans ce travail tel que les ANN, SVM, les decision tree, les algorithmes génétiques, le K-means clustering et bien d'autres. Concernant les points ressortant de cette analyse ayant un intérêt dans ce mémoire on soulignera principalement que les ANNs sont les plus performants que les SVM (SVM prenant un temps de calculs plus longs ce qui impacte la performance générale de cette méthode fortement) dans plusieurs domaines notamment dans la prédiction de la quantité d'énergie requise. Ce résultat est dû à la prédisposition des ANNs à gérer les problèmes non-linéaires comme énoncé précédemment.

En Octobre 2017 un groupe de recherche composé de personnes provenant de l'école d'information et d'ingénierie électrique de l'université de Shandong Jianzhu ainsi que de l'institut d'automation de l'académie des sciences de Pékin à quand à lui créé un système utilisant des SAEs (Stacked Auto Encoders) ainsi que l'ELM (Extreme Learning Machine) afin de prédire la consommation d'un bâtiment avec plus de précision que les méthodes précédemment explorées de part la combinaison de plusieurs techniques de machine learning [Li+17]. Dans leurs système les SAEs ont servi à extraire les features qui allaient ensuite être utilisées afin de réaliser les prédictions grâce à la partie ELM du système combinée à une analyse de corrélation partielle via la PACF (Partial AutoCorrelation Function) qui seront discutés plus tard dans ce mémoire. Afin de comparer la performance de cette manière de procéder les résultats ont été mis en parallèle avec des algorithmes tel que ceux cités précédemment et les résultats tendent à indiquer que de par son architecture plus profonde que les autres cette technique donne de meilleurs résultats et nécessiterait

d'être approfondie avec notamment l'utilisation d'un jeu de donnée plus grand.

En 2018 un groupe de chercheur provenant de l'université Pablo Olavide en Espagne et de l'université de Namur en Belgique ont travaillé sur une approche de la prédiction de la consommation électrique sur le court terme via l'utilisation de l'ensemble learning [Div+18]. L'ensemble learning consiste en la combinaison de plusieurs techniques machine learning en une seule afin d'obtenir des résultats plus performants que si ces techniques étaient utilisées individuellement. Pour utiliser ces multiples prédictions il existe plusieurs façons de faire comme le "bagging" qui peut s'apparenter à un vote avec une importance égale de chaque voteur ou inégale dans le cas du "boosting". Dans ce travail la méthode choisie est le "stacking" qui consiste lui contrairement à celles énoncées avant à ne pas choisir la meilleure prédiction mais à combiner toutes celles produites en une seule grâce à un algorithme. Les modèles choisis ici pour réaliser les prédictions individuelles sont : regression tree basé sur les Evolutionnary Algorithm ou EVTree, RF, et ANN. Ces trois prédictions sont ensuite combinées grâce à un Generalized Boosted Regression Model (GBM). Les résultats montrent que l'utilisation de l'ensemble learning donne des résultats plus satisfaisants que les autres méthodes individuelles mais souligne aussi le fait que les réseaux des neurones se démarquent aussi du lot et plus particulièrement sur le court terme ce qui est un résultat intéressant pour le travail qui sera détaillé plus bas.

En Mai 2019 la même équipe à l'exception d'un membre de l'université Pablo Olavide a cette fois réalisé en partenariat avec un membre de l'université américaine du Paraguay un travail de comparaison de plusieurs techniques de machine learning dans le cadre de la prédiction de la consommation d'un "smart" bâtiment (donc un bâtiment doté de smart meters mais ici équipé aussi de capteurs surveillant des variables telles que l'utilisation de l'air conditionné, du chauffage ou encore de la lumière) [Div+19]. Les techniques comparées dans ce document sont les suivantes : LM (Linear Model) ou aussi appelé régression linéaire, ARIMA, EVTree,GBM (étant une technique d'ensemble learning elle sera ici utilisée avec un ensemble de regression tree et un mécanisme de vote), ANN, RF, la méthode développée dans l'article ci-dessus avec l'ensemble learning, le "Recursive Partitioning and Regression Trees" (RPart) et l'Extreme Gradient Boosting (XGBoost). Les résultats ont montré que dans ce cas-ci les algorithmes RF et GBM donnaient les meilleurs résultats avec le taux d'erreur le plus bas sans se démarquer fortement de XGBoost, LM et ANN pour autant.

En Septembre 2019 Sachin Bhoite à quant à lui a réalisé un travail de prédiction de la consommation d'une maison individuellement en utilisant pour ce faire le modèle ARIMA afin d'identifier les éventuels patterns sur différents intervalles de temps [PB19]. Le nombre de trous présents dans le

dataset étant assez consistant ceux-ci ont été remplis à l'aide d'une fonction assez simple reprenant la valeur présente 24h auparavant. Une fois ce dataset complet une analyse a été appliquée à différentes échelles et ont montré que le modèle ARIMA se prêtait le mieux pour des prédictions de l'ordre d'un mois, un trimestre, une journée et une semaine.
De tous ces travaux sont ressortis que la prédiction de la consommation électrique ne semble pas avoir de méthode parfaitement prédisposée à réaliser cette tâche néanmoins certaines d'entre elles semblent malgré tout faire mieux que la moyenne. Les réseaux de neurones, qui selon plusieurs travaux se prêtent particulièrement bien aux problèmes non-linéaires et sur des prédictions à court terme constituant la plus intéressante d'entre toutes. Les ANNs seront donc étudiés ainsi que RF qui semble avoir de bons résultats aussi. Ce mémoire parcourra donc ces méthodes afin d'éventuellement déterminer si l'une d'entre elles se prête mieux à l'échelle qu'est celle de ce projet, c'est-à-dire au niveau de la consommation d'un groupement de plusieurs habitations résidentielles et au cours d'une période définie allant d'une semaine à 6 mois.

3. Développement de la recherche

3.1 Données

Comme expliqué dans l'introduction le data-set est fourni par la firme My electric avenue [Ave] et comprend plusieurs fichiers Excels comme le montre ci-dessous la figure 3.1.

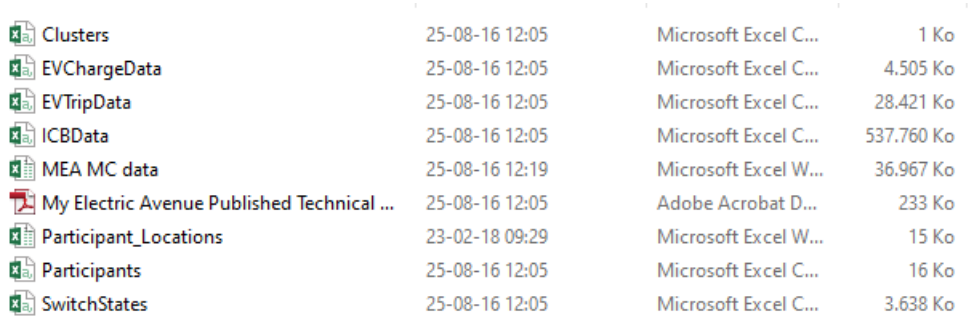

FIGURE 3.1 : Les données fournies

Comme on peut le voir ce dossier semble contenir pas mal d'informations ainsi que des fichiers allant jusque 500Mo. Le fichier PDF "My Electric Avenue Published Technical Data Overview" contient lui la description de ce que représente ces différents fichiers malheureusement celle-ci étant très mal documentée seul le fichier "MEA MC Data" sera exploité durant ce travail. Malgré cela voici une brève description du contenu de chaque fichier :

- `Clusters.csv` : contient des informations sur les différentes populations étudiées (appelée ici cluster) telles que leurs préfixes, l'identifiant du feeder (câble fournissant l'énergie au cluster), le nombre de maisons branchées sur le feeder ou encore le nombre de véhicules électriques que possède ce cluster.
- `EVChargeData.csv` : contient des données sur les temps de chargement des véhicules électriques avec l'identifiant du véhicule et des données tel que le moment où sa charge a démarrée et quand s'est-elle terminée.

- `EVTripData.csv` : contient cette fois-ci des données sur l'utilisation des véhicules électriques et les trajets effectués avec ceux-ci. On retrouve donc les identifiants ainsi que la distance parcourue, la consommation due à ce trajet et l'intervalle de temps durant lequel cela s'est produit.

- `ICData.csv` : le contenu de celui-ci reste flou mais selon la courte description de My electric avenue ces données sont relatives à un boîtier que la firme a installé sur les véhicules électriques afin de couper la recharge de ceux-ci selon leurs volontés. Le document contient donc les identifiants des véhicules ainsi que les intervalles où la charge de ceux-ci a été interrompue. Après un rapide passage en revue de ces données il s'est avéré qu'approximativement 95% de ces temps de coupure étaient non-évalués (l'intervalle de temps étant [NULL,NULL], NULL représentant l'absence de valeur) rendant donc ces données inutilisables.

- `MEA MC Data.xls` : celui-ci faisant l'objet de la recherche il sera détaillé par après plus en détail.

- `Participant_Location.csv` : contient le lien entre chaque identifiant de participant (donc le préfixe du cluster auquel il appartient et son chiffre personnel) et ce qui serait apparemment sa location sous forme d'identifiant mais ce fichier n'étant même pas abordé dans le PDF cité plus haut cette description s'avère n'être que des suppositions.

- `Participants.csv` : contient les identifiants des participants (donc le préfixe de son cluster et son numéro personnel accolé à celui-ci) ainsi que des données peu explicites et surtout le moment où celui-ci a rejoint l'expérience et l'a ensuite quittée si tel s'est produit.

- `SwitchStates.csv` : contient les identifiants des participants possédant un véhicule électrique ainsi que le moment ou la charge de leurs véhicules a été changée c'est-à-dire est passée de ON à OFF ou inversement.

Le document MEA MC Data contient comme on peut le voir sur la figure 3.2 premièrement le nom complet du cluster (MC Name) qui est concerné, le numéro de série du feeder (câble d'alimentation fournissant l'énergie à l'entièreté du dit cluster), la timestamp où cette observation a été prise (la suite de ces timestamps prises à intervalle régulier associées à leurs valeurs forment donc ce que l'on appelle une série temporelle) et la demande en énergie électrique sur chacune des trois phases du feeder appelés respectivement P1, P2 et P3. Concernant les intervalles entre les différentes timestamps ceux-ci seront détaillés dans la partie pré-processing car une opération de tri a dû être réalisée. L'unité de mesure de la demande n'étant pas spécifiée aucune unité ne sera donc utilisée.

1	MC Name	MC Serial Number	timestamp	P1 Current	P2 Current	P3 Current
2	Chineham - Cufaude Village	14778	39:00,0	NULL	NULL	NULL
3	Chineham - Cufaude Village	14778	00:00,0	NULL	NULL	NULL
4	Chineham - Cufaude Village	14778	53:00,0	31,33	62,72	68,23
5	Chineham - Cufaude Village	14778	00:00,0	26,36	44,78	44,87
6	Chineham - Cufaude Village	14778	10:00,0	45,36	50,74	41,7
7	Chineham - Cufaude Village	14778	20:00,0	32,33	43,49	36,5
8	Chineham - Cufaude Village	14778	30:00,0	41,79	70,04	40,54
9	Chineham - Cufaude Village	14778	40:00,0	40	85,17	39,36
10	Chineham - Cufaude Village	14778	50:00,0	38,24	60,31	37,04
11	Chineham - Cufaude Village	14778	00:00,0	16,07	48,73	34,24
12	Chineham - Cufaude Village	14778	10:00,0	32,38	72,38	52,92
13	Chineham - Cufaude Village	14778	20:00,0	45,87	61,34	49,66
14	Chineham - Cufaude Village	14778	30:00,0	40,34	89,19	51,42
15	Chineham - Cufaude Village	14778	40:00,0	43,96	56,3	54,36
16	Chineham - Cufaude Village	14778	50:00,0	53,85	43,39	48,94
17	Chineham - Cufaude Village	14778	00:00,0	40,79	66,39	42,8
18	Chineham - Cufaude Village	14778	10:00,0	51,61	50,64	32,2
19	Chineham - Cufaude Village	14778	20:00,0	57,71	55,55	31,32
20	Chineham - Cufaude Village	14778	30:00,0	36,71	72,06	31,99
21	Chineham - Cufaude Village	14778	40:00,0	55,02	53,11	42,43
22	Chineham - Cufaude Village	14778	50:00,0	40,2	79,09	28,71
23	Chineham - Cufaude Village	14778	00:00,0	28,65	82,27	36,09

FIGURE 3.2 : MEA MC Data

La demande totale en énergie représentant la somme des 3 demandes de chaque phase [Max] ce qui est intéressant dans ce mémoire c'est de pouvoir prédire quand celle-ci dépassera un certain seuil afin que le fournisseur puisse mettre en place une solution et éviter la surcharge du feeder. Le seuil maximum du feeder n'étant pas fourni le travail a été réalisé en fixant celui-ci de manière arbitraire afin de simuler l'utilisation en temps réel des prédictions produites par le modèle comme expliqué dans la section résultats du mémoire.

3.2 Pre-processing des données

Dans le fichier excel MEA MC Data se trouve donc une multitude d'observations concernant uniquement 3 des 12 clusters étudiés par la firme : Chineham (identifié par CRG), Chiswick (CC), Lyndhurst (BL). Les observations des 3 clusters étant mélangées dans le même document et le temps d'intervalle entre 2 observations n'étant pas constant celui-ci variant entre 1 et 10 minutes il faut d'abord trier celles-ci.
La première étape à réaliser avant de pouvoir travailler sur ces données est de récupérer les observations relatives aux différents clusters et ensuite de les trier par temps d'écart entre les observations afin de ne pas mélanger des données espacées de 10 minutes avec d'autres espacées de seulement 1 minute. Afin de tirer parti au maximum de cette diversité dans les données celles-ci ont été séparées en fonction de la période étudiée (respectivement ici allant de une semaine à 6 mois selon la quantité de données disponibles sur le cluster) tout en sachant que les périodes plus petites sont des sous-ensembles de celles relatives au même cluster pour que les clusters ne possédant pas toujours le même nombre de données aient une échelle commune afin d'être comparés.
Le résultat de ces découpes est le suivant :

1. Un fichier Excel sur le cluster BL durant une période de une semaine avec un espacement de une minute
2. Un fichier Excel sur le cluster CC durant une période de une semaine avec un espacement de dix minutes
3. Un fichier Excel sur le cluster CC durant une période de un mois avec un espacement de dix minutes
4. Un fichier Excel sur le cluster CRG durant une période de une semaine avec un espacement de une minute
5. Un fichier Excel sur le cluster CRG durant une période de une semaine avec un espacement de dix minutes
6. Un fichier Excel sur le cluster CRG durant une période de un mois avec un espacement de dix minutes
7. Un fichier Excel sur le cluster CRG durant une période de trois mois avec un espacement de dix minutes
8. Un fichier Excel sur le cluster CRG durant une période de six mois avec un espacement de dix minutes

Les données disponibles ne faisant jamais exactement la période voulue la découpe de ces périodes a été favorisée sur les données contenant le moins

de trous possibles afin d'éviter au maximum le besoin d'approximer ces valeurs manquantes. Néanmoins il restait malgré tout certaines valeurs vides pour lesquels la méthode utilisée par Mr Bhoite dans le travail [PB19] a été appliquée en reprenant la valeur de cette même période 24h auparavant.

3.2.1 Découpe utilisée pour le réseau de neurones et la random forest

Après avoir choisi le cluster et la période de celui-ci que l'on souhaite utiliser les données sont d'abord séparées entre la partie servant à l'entraînement du modèle et la partie qui sera utilisée pour vérifier la précision du modèle. Dans le cadre de ce mémoire le nombre de prédictions à réaliser a été arbitrairement fixé à 120 observations ce qui représente 2 heures dans le cadre d'un suivi des données toutes les minutes et 20 heures pour un espacement de 10 minutes ce qui sont des périodes que l'on peut qualifier de court-terme comme on le souhaite. Les données restantes seront affectées à l'entraînement du réseau de neurones.
Dans le cadre de l'utilisation d'un réseau de neurones ou d'une random forest avec des données sous la forme de séries temporelles il faut d'abord découper les données de manière à pouvoir passer au modèle une série de x observations consécutives afin qu'il fournisse selon lui quelle valeur devrait apparaître par la suite. Cette façon de fonctionner sera expliquée plus en détail dans la section dédiée aux méthodes utilisées mais l'idée fondamentale est d'organiser la série en plusieurs sous-séries afin de pouvoir entraîner le modèle à reconnaître par le biais du machine learning comment construire des prédictions les plus précises possibles. Afin de pouvoir passer l'entièreté des données d'entraînement au modèle il faut organiser celles-ci sous la forme d'une matrice où chaque colonne serait une sous-série de taille fixée (nos x observations consécutives) ce qui donne une matrice ressemblant à celle-ci :

$$M = \begin{bmatrix} t & t+1 & .. & .. \\ t+1 & t+2 & .. & .. \\ .. & .. & .. & .. \\ t+x & t+x+1 & .. & t+n \end{bmatrix}$$

Durant cette découpe qui est effectuée par la fonction que l'on peut voir sur la figure 3.3 celle-ci va stocker aussi dans la variable DataY la valeur qui est censée suivre la sous-série stockée dans DataX. Lors de l'entraînement le modèle va donc évaluer une sous-série de DataX et produire un résultat qui sera ensuite comparé à la réponse correcte se trouvant dans DataY et ainsi corriger son fonctionnement en fonction de son erreur comme on le verra dans la partie sur les méthodes utilisées.

```python
def create_dataset(df, previous=1):
    dataX, dataY = [], []
    for i in range(len(df)-previous-1):
        a = df[i:(i+previous), 0]
        dataX.append(a)
        dataY.append(df[i + previous, 0])
    return np.array(dataX), np.array(dataY)
```

FIGURE 3.3 : La fonction permettant le découpage des données en vue de l'entraînement du modèle

3.3 Analyse des propriétés des séries temporelles

Les données comme expliqué dans la découpe expliquée précédemment sont organisées sous la forme d'une suite de valeurs séparées par un intervalle de temps de soit 1 soit 10 minutes et sur des durées allant de une semaine à 6 mois. Les séries concernant la même population et le même intervalle étant les mêmes sur des durées différentes (ayant pour unique but la comparaison des performances en fonction de la quantité de données disponibles), seules les plus grandes seront analysées dans cette section. Les points qui sont importants de cette section seront majoritairement utilisés dans le cadre du modèle ARMA et ses dérivés ARIMA et SARIMA qui seront vus dans la section suivante. Tout d'abord une brève vue d'ensemble des séries de manière générale est présentée sur les figures 3.4, 3.5, 3.6 et 3.7 afin de voir comment celles-ci se comportent et ainsi mieux visualiser le fonctionnement des séries temporelles avant de les décortiquer. Tout d'abord il faut savoir qu'une time-series est composée de 4 éléments : le niveau (ou la valeur moyenne de la série), la tendance (valeur dont la série augmente/diminue de manière constante dans le temps), la saisonnalité (cycle se répétant dans les données sur une période de temps inférieure à un an) et le bruit (variation aléatoire) [Bro]. Maintenant en passant en revue chacune des figures et en regardant après les composants cités ci-dessus le constat et le suivant :

1. Le niveau semble être plus ou moins le même sur les figures représentant les villes de Lyndhurst et Chineham mais diffère fortement de celui de Chiswick.

2. Aucune des séries présentées ne semble être sujette à une tendance.

3. Toutes les figures à l'exception de la numéro 3.6 semblent posséder un cycle se répétant et sont donc sujettes à la saisonnalité. De plus

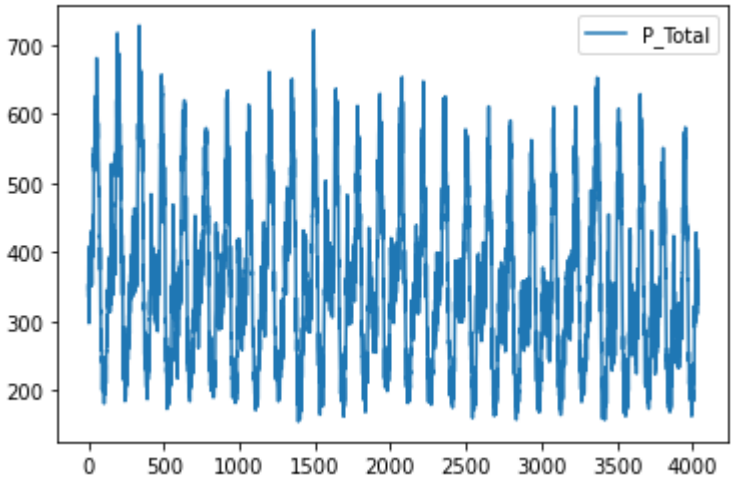

FIGURE 3.4 : Chiswick avec un intervalle de 10 minutes sur une durée d'un mois

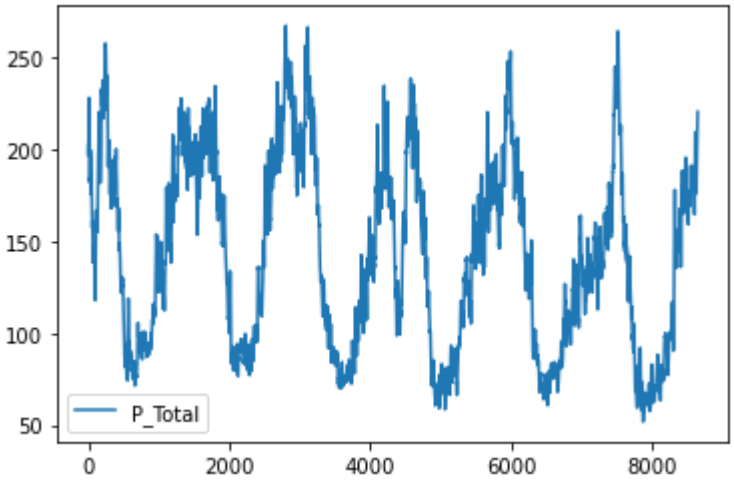

FIGURE 3.5 : Lyndhurst avec un intervalle de 1 minutes sur une durée d'une semaine

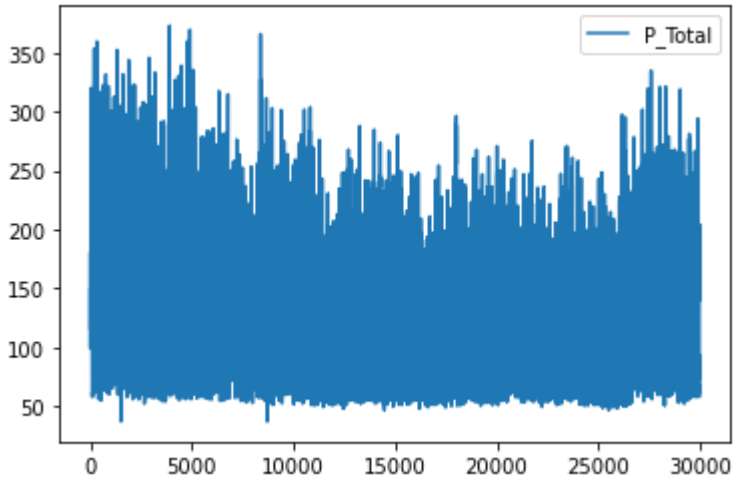

FIGURE 3.6 : Chineham avec un intervalle de 10 minutes sur une durée d'un mois

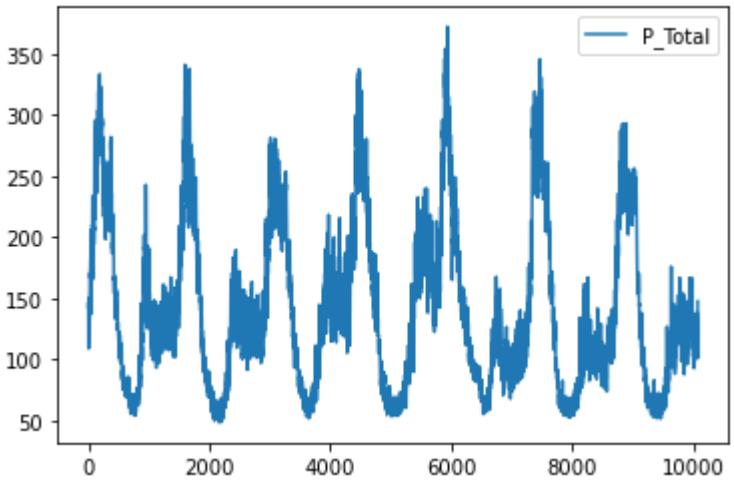

FIGURE 3.7 : Chineham avec un intervalle de 1 minute sur une durée d'une semaine

ce cycle si l'ont regarde bien représente une journée complète soit 144 observations dans le cas de la figure 3.4 (et 3.6 même si cela n'est pas très visible vu la densité du graphique) et 1440 pour les deux autres.

4. Toutes les séries semblent être sujettes au bruit.

Par définition une série temporelle est stationnaire si : sa moyenne (ou niveau) est constante , son écart-type est constant (cela semble être le cas partout aussi sauf peut-être dans le cas de la figure 3.6) et il n'y a pas de

saisonnalité, ce qui n'est pas le cas ici [Rit]. Le fait de ne pas être stationnaire implique que certains modèles tel que ARMA ne sont pas applicables à moins de transformer ces données et rendre la série stationnaire.

Une deuxième chose à laquelle regarder maintenant sont les graphes d'auto-corrélation et de corrélation partielle que l'on obtient de ces données. Le graphe d'auto-corrélation représente le lien de corrélation entre une observation et celles la précédent. Chaque barre sur le graphique de gauche de la figure 3.8 représente l'effet direct qu'a l'observation au temps $t - x$ (x étant l'axe des abscisses valant 1,2,3,... et s'appelant "lag") sur l'observation au temps t mais prend aussi en compte l'effet indirect des autres lags : si la valeur au temps t est influencée par celle au temps $t - 1$ mais que cette valeur en $t - 1$ subit elle-même l'influence de celle au temps $t - 2$ alors la valeur en $t - 2$ influence indirectement celle au temps t. Le lag 5 sur ce graphe représente donc l'effet de l'observation au temps $t - 5$ sur celle au temps t en prenant en compte les effets directs et indirects des autres lags agissant sur $t - 5$. Le graphe d'auto-corrélation partielle lui sert à mesurer uniquement l'effet direct de ces lags.

Malheureusement comme écrit précédemment les séries ne sont pas stationnaires et leurs graphes de corrélation totale et partielle s'en retrouvent fortement impactés ce qui fait que tous ressemblent fortement à ceux de la figure 3.8. Le fait d'avoir autant de lags au-dessus de cet intervalle bleu, signifiant grossièrement que les lags situés à l'intérieur peuvent être évalués à 0, est un signe que la série n'est pas stationnaire ce qui confirme l'observation précédente. Afin donc de la rendre stationnaire il faut supprimer la saisonnalité précédemment identifiée. Pour ce faire il faut recourir à une différenciation : l'observation au temps t doit être soustraite à celle un cycle plus loin ($t-$`Taille du cycle`). Cette opération est réalisée par la fonction présente sur la figure 3.9.

Sur la figure 3.10 les données de la figure 3.4 ont été différenciée et comme on peut le voir il ne semble plus y avoir de présence de cycle dans celles-ci. Malheureusement cela n'a pas résolu totalement le problème comme la figure 3.11 le montre, c'est pourquoi afin de tenter de régler le problème une idée est de procéder à une deuxième différenciation cette fois-ci non pas entre les valeurs à un intervalle d'un cycle mais uniquement à un intervalle de une observation (qui est la valeur par défaut de la fonction utilisée). Seul les deux premières valeurs sortent de l'intervalle ce qui est bon signe mais la deuxième barre descend trop rapidement vers les négatifs ce qui signifie que la série est probablement "sur-différenciée" [plu] donc la précédente sera favorisée à celle-ci. À noter que le graphe d'auto-corrélation de la figure 3.11 contient plus de barres que celui de la figure 3.12 et ce car par soucis de clarté, la figure 3.11 nécessitait plus de lags afin de mettre en avant le problème qui se posait.

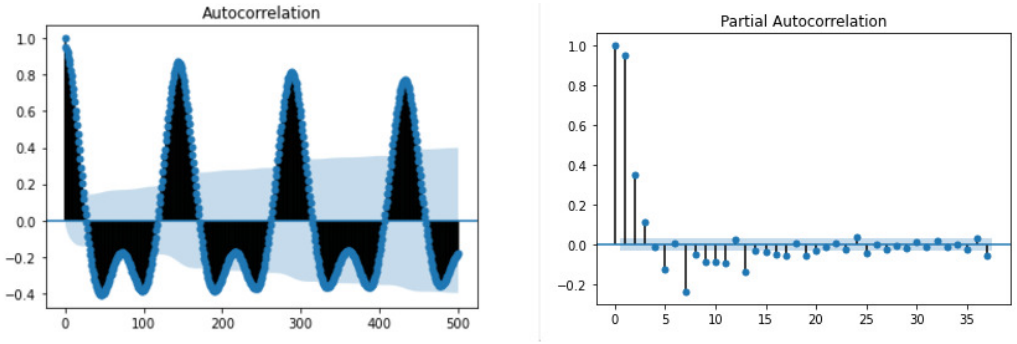

FIGURE 3.8 : Chiswick ACF et PACF sur un mois avant différenciation

```
def difference(dataset, interval=1):
    diff = list()
    for i in range(interval, len(dataset)):
        value = dataset.get(i) - dataset.get(i - interval)
        diff.append(value)
    return Series(diff)
```

FIGURE 3.9 : Fonction utilisée pour supprimer la saisonnalité

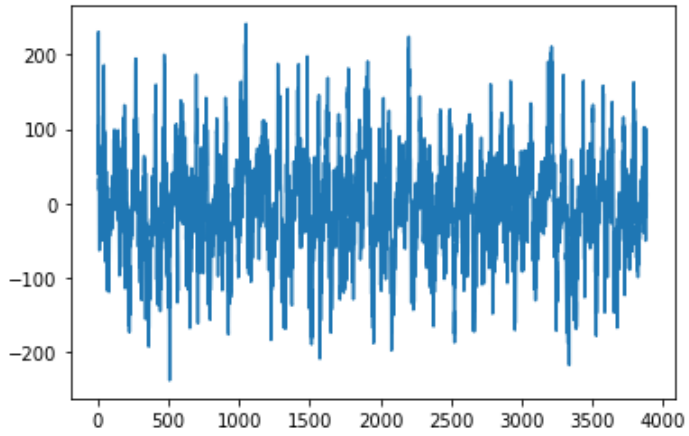

FIGURE 3.10 : Chiswick sur un mois avec 10 minutes d'intervalle après différenciation

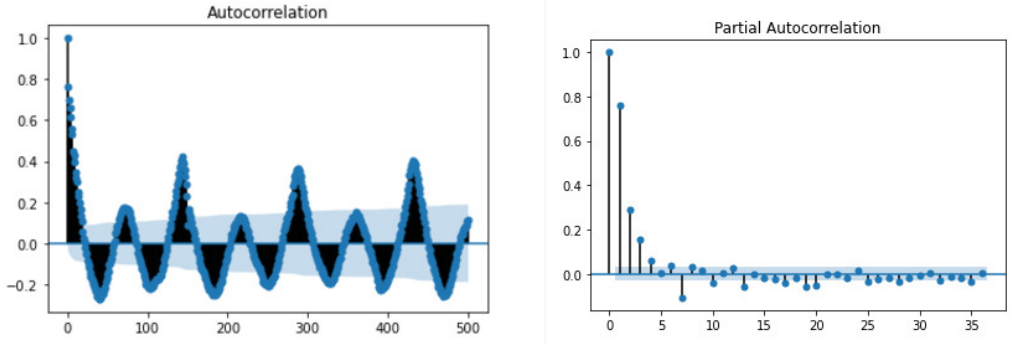

FIGURE 3.11 : Chiswick ACF et PACF sur un mois après différenciation

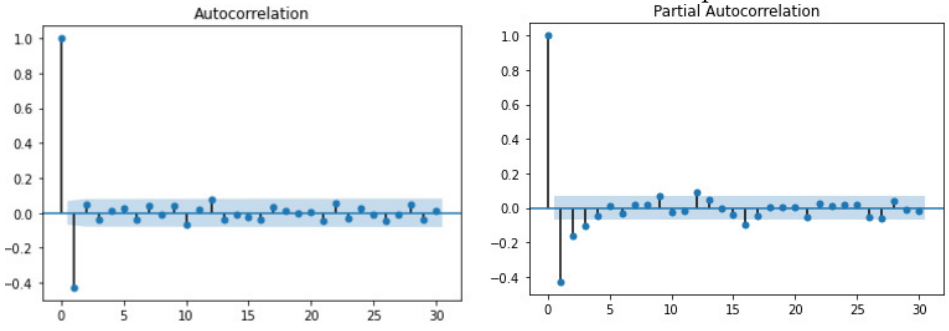

FIGURE 3.12 : Chiswick ACF et PACF sur un mois après les deux différenciations

3.4 Méthodes utilisées

Dans cette section les différentes approches mises en place dans le cadre de la prédiction des séries temporelles vont être passées en revue pour voir comment celles-ci ont été appliquées dans ce projet.
ARMA et ARIMA étant empiriquement moins efficaces que les autres méthodes comme la partie état de l'art le montre celles-ci vont servir de point de départ pour comparer les performances des autres. SARIMA quant à lui sera considéré à part entière même si il s'agît d'une version améliorée d'ARMA comme l'explique la section suivante.

3.4.1 ARMA, ARIMA et SARIMA

Les séries temporelles (ou time series) représentant la valeur prise par une variable à un temps donné les modèles ARMA, ARIMA ainsi que SARIMA proposent chacun une équation permettant d'approcher au mieux le comportement de celle-ci [Whi51]. Pour mieux comprendre comment ces équations tentent de répliquer le comportement de la time-series un rapide passage sur les fondements mathématiques de celles-ci va être réalisé.
Tout d'abord ARMA ou autoregressive moving average est la combinaison de deux modèles assez simples qui s'appellent autoregressive (AR) et moving average (MA).
AR s'écrit sous la forme :

$$X_t = \text{constante} + \beta_1 X_{t-1} + \beta_2 X_{t-2} + ...\beta_p X_{t-p} + \epsilon_t$$

Et MA sous la forme :

$$X_t = \mu + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + ... + \phi_q \epsilon_{t-q}$$

Où X_t est la valeur de la time-series au moment t, β_n est le coefficient qui sert à quantifier l'impact direct de la n-ième valeur sur X_t, ϵ_t est l'erreur entre la valeur donnée par le modèle au temps t et la valeur réel au même temps, μ est la moyenne (average de MA) et ϕ_q est quand à lui le coefficient permettant de prendre en compte l'erreur au temps $t - q$ pour approximer au mieux la nouvelle prédiction d'où le nom "moving average" car la moyenne est corrigée selon les erreurs précédentes. En remplaçant μ par l'équation de AR et en réunissant les ϵ_t en un seul l'équation de ARMA s'écrit :

$$X_t = \beta_0 + \beta_1 X_{t-1} + \beta_2 X_{t-2} + ... + \beta_p X_{t-p} + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + ... + \phi_q \epsilon_{t-q}$$

Cette équation est bien entendu la forme la plus générique de celle-ci et c'est pourquoi elle possède deux paramètres p et q qui définissent

respectivement le degré de la partie AR et MA du modèle. Ces degrés sont calculés via la fonction d'auto-corrélation partielle (PACF) pour trouver p et d'auto-corrélation (ACF) pour q qui ont été vus dans la section précédente. Dans les deux graphiques les lags se situant au delà de l'intervalle signifient que le lag est à prendre en compte dans le calcul de la nouvelle valeur. Malheureusement ce cas est faussé et impliquerait que les x premiers lags seraient à prendre en compte et ensuite les y autres quelques lags plus tard et ainsi de suite ce qui n'est pas réalisable sans écrire l'équation manuellement et évaluer celle-ci. Dans un cas stationnaire les n premiers lags auraient dépassé l'intervalle et les suivants se situeraient dans celui-ci n'entraînant pas de discontinuité dans la suite des lags à prendre en compte. Les degrés p et q ne seront donc pas calculés manuellement mais seront approximer par une fonction automatique de la librairie Pyramids [Pyr]. Cette fonction prend en argument les données et va tester l'ensemble des combinaisons de p et q possibles avant de trouver celle optimale grâce au critère d'inférence Bayésien [Sch78]
ARIMA quant à lui n'est rien d'autre qu'un modèle ARMA auquel on a rajouté la partie Integrated (I de ARIMA) qui permet, lorsque la série temporelle contient une tendance (donc semble augmenter de manière constante d'une certaine valeur) de mieux approximer le comportement de celle-ci. D'un point de vue mathématique pour réaliser cela on opère à une différenciation :

$$Y_t = X_{t+1} - Xt$$

Où Y remplacera X dans l'équation :

$$X_t = \beta_0 + \beta_1 X_{t-1} + \beta_2 X_{t-2} + ... + \beta_p X_{t-p} + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + ... + \phi_q \epsilon_{t-q}$$

Le degré de différenciation d définira le nombre de fois que cette substitution devra être faite pour obtenir une time series sur laquelle on pourra appliquer ARMA par après. Les fonctions finales s'écriront donc $ARMA(p, q)$ et $ARIMA(p, d, q)$. Sur la figure 3.13 se trouve un exemple où la série possède une tendance et un modèle ARIMA(5,1,0) a été entraîné pour prédire les valeurs en rouge sur la partie de droite de l'image. Cet exemple est tiré du site Machine learning mastery [masb].
SARIMA quant à lui va donc rajouter la prise en compte de la saisonnalité (S pour seasonal) au modèle ARIMA et effectuer non pas comme il a été discuté auparavant une différenciation mais un ajout de paramètres permettant la suppression de ce comportement cyclique pour rendre la série temporelle stationnaire. L'équation étant fort complexe et non-nécessaire à la compréhension de ce travail elle ne sera pas détaillée dans ce travail. Cependant il est intéressant de regarder à la forme que va prendre la nouvelle formule car la librairie utilisée [Pyr] ne va pas opérer à une différenciation mais va rajouter des composants à l'équation d'ARIMA

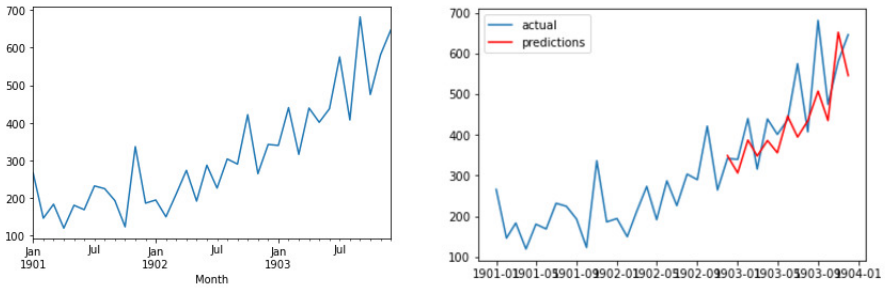

FIGURE 3.13 : Un exemple de prédiction avec un modèle ARIMA(5,1,0)

pour approximer le comportement de cette saisonnalité. La nouvelle formule est donc : $ARIMA(p,d,q)(P,D,Q)_m$, où m est la taille du cycle et (P,D,Q) représentent les mêmes termes que ceux de ARIMA mais dans le cadre de la saisonnalité. Grâce à la fonction automatique de Pyramids seul m est à fournir à celle-ci car les combinaisons des autres $((p,d,q)et(P,D,Q))$ seront toutes testées afin de trouver la combinaison la plus efficace de ces termes.

3.4.2 LSTM

Comme l'état de l'art en a déjà un peu parlé le modèle LSTM est un type particulier de réseau de neurones dont les propriétés sont plus adaptées au cas d'étude. Le modèle LSTM est une version améliorée des réseaux de neurones dit "feed-forward" qui eux ne permettent pas de travailler avec des séries temporelles. Cette partie discute des différentes améliorations apportées par LSTM à ceux-ci [Gre+17].

Un réseau de neurones artificiel (ou ANN) est un système d'apprentissage cherchant à imiter le comportement d'un cerveau humain et pour ce faire celui-ci tout comme le cerveau est composé d'unités appelées neurones fortement inter-connectées. Dans le cadre d'un ANN les neurones vont être répartis comme la figure 3.14 l'illustre en couches de un ou plusieurs neurones connectés à ceux des couches voisines. Un neurone va donc recevoir un certain nombre d'entrées de la couche précédente et grâce à une fonction d'activation propre au neurone celui-ci va produire une valeur ou non.

Comme la figure 3.14 l'illustre il existe 3 catégories de couches à savoir la couche d'entrée, où fournir nos données à calculer, une ou plusieurs couches dites cachées servant au traitement de ces données et une dernière couche de sortie où l'on obtient le résultat de ce traitement. Dans le réseau de neurones le plus simple appelé le "feedforward" va durant la phase d'entraînement assigner un poids (w) à chaque lien et ce poids servira à définir si la fonction d'activation du neurone recevant doit se déclencher ou

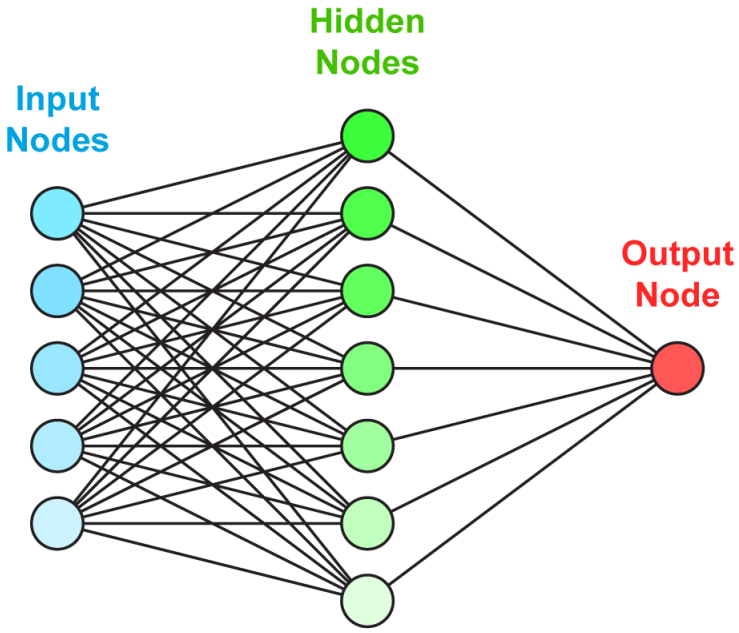

FIGURE 3.14 : Un exemple simple de réseau de neurones [Smi]

non. Soit le cas de la figure 3.14 et avec la valeur d'entrée du premier neurone vert définie comme v_1 et les sorties des 5 neurones d'entrées comme $b_i, i \in [1, 2, 3, 4, 5]$. L'équation ressemblera donc à :

$$v_1 = \sigma(\sum_{i=1}^{5} w_{i_v 1} b_i)$$

Où σ est la fonction d'activation valant 1 si l'input est suffisant et 0 sinon, $w_{i_v 1}$ est le poids attribué au lien entre le neurone d'entrée i et le neurone vert numéro 1. Si la fonction σ dépasse donc le seuil d'activation fixé alors le neurone calculera une valeur v_1 qu'il distribuera à son tour aux neurones de la couche suivante. Durant le processus d'apprentissage la valeur produite sera donc comparée à la valeur réellement attendue et les poids seront changés pour corriger au mieux l'erreur [War43].
Dans le cadre d'un modèle LSTM plusieurs éléments viennent se rajouter à cette version de base. Premièrement il s'agît d'un réseau de neurones récurrent ce qui implique un système de feedback intégré au modèle via le fait que les neurones d'un RNN (Recurrent Neural Network) [RHW86] sont connectés à eux-mêmes mais aussi à ceux de la même couche ce qui permet de travailler avec des séquences de données comme par exemple une série temporelle. Cette particularité permet de faire des prédictions de valeurs

contrairement au feedforward qui lui sera plutôt utilisé pour réaliser par exemple un travail de classification ou de régression. Maintenant le problème d'une telle approche est que l'entraînement d'une couche avec une longue série de données sera sujet au problème dit du "vanishing gradient". Ce problème empêche la correction efficace des poids associés aux connexions lorsque le réseau de neurones est trop long car elle devient infime au fur et à mesure que l'on revient en arrière ce qui rend ce processus moins efficace [KK01]. Pour palier à ce problème des RNN le système de "gate" a été créé et permet au système d'évaluer à quel moment oublier la valeur d'entrée courante et quand la retenir pour une utilisation future. Sur la figure 3.15 se trouve la structure d'une unité d'un réseau LSTM avec 3 gates agissant chacune comme un neurone d'un réseau feedforward avec une fonction d'activation. Voici pour chacune d'entre elles le comportement adopté par l'unité de calcul :

1. Forget gate : Sa fonction d'activation va définir l'information précédemment stockée qui n'a plus d'intérêt au calcul de celle qui est actuellement traitée et doit être oubliée.

2. Input gate : Va mettre à jour l'information de l'unité avec celle qui vient d'entrer dans le système.

3. Output gate : Va définir la valeur de sortie de l'unité.

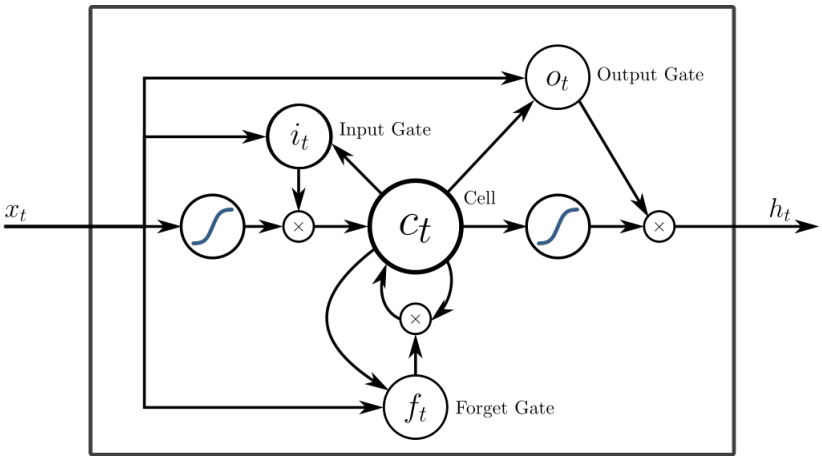

FIGURE 3.15 : La structure d'une unité de calcul d'un modèle LSTM [Wik]

Pour mieux comprendre le déroulement d'une étape dans le modèle LSTM (et plus particulièrement celui de la librairie keras [Ker] qui est celle utilisée dans le projet) un rapide passage en revue des mécanismes mis en place lors d'une itération va être fait. Sur la figure 3.16 se trouve l'architecture d'une cellule typique utilisée dans ce projet. Les valeurs C_t et

h_t représentent respectivement l'état actuel de la cellule (l'information qu'elle a stockée depuis qui constitue la partie 1 en rouge) et l'information qu'elle traite et qui passe par la gate d'oubli (partie 2 en bleu), la gate d'input (partie 3 en orange) et la gate d'output (partie 4 en vert) avant de ressortir de la cellule.

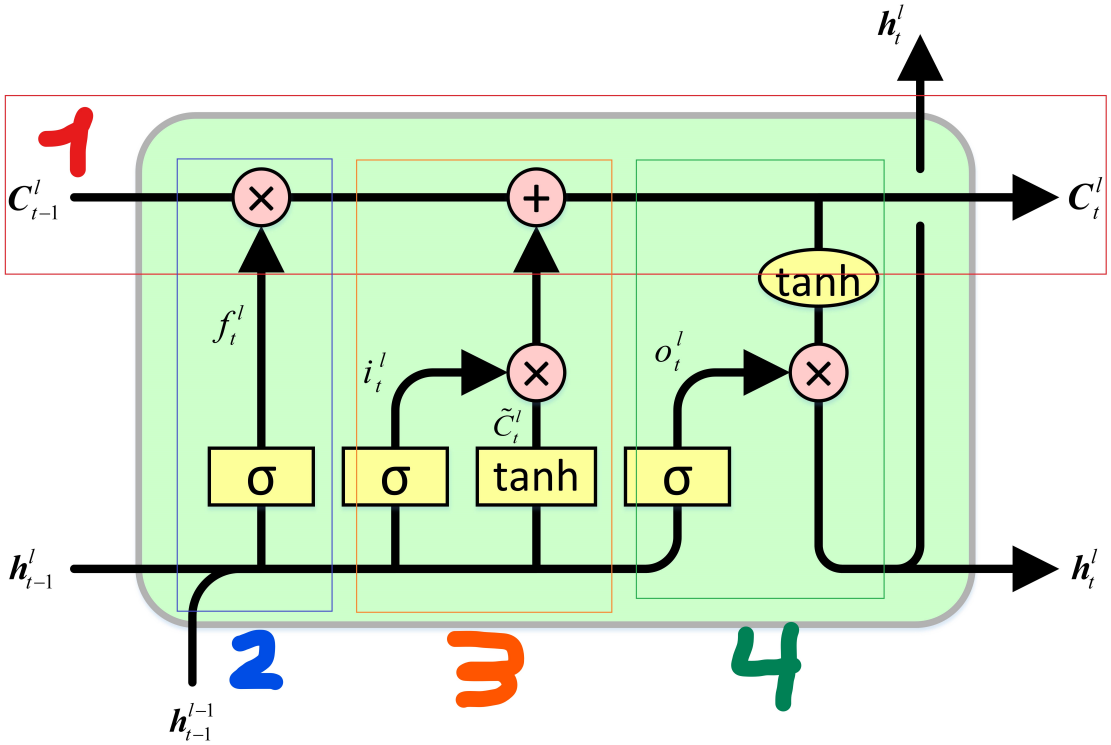

FIGURE 3.16 : L'architecture d'une cellule LSTM de la librairie keras, modifiée depuis [Sta]

Les σ représentent les fonctions d'activation déterminant la quantité de donnée devant être oubliée, ajoutée à l'état actuel ou devant sortir de la cellule et celles-ci combinées à des opérations d'addition, multiplication ou tanh forment l'architecture des cellules du réseau LSTM [Gre+17].

3.4.3 Random forest

La technique du random forest est assez facile à comprendre tout en étant un outil puissant combinant par le biais de l'ensemble learning explicité dans l'état de l'art une multitude de prédictions données par des modèles moins puissants et en les combinant à l'aide d'une technique de boosting comme vu dans [Div+18]. Ces multiples prédictions sont créées à partir d'arbres de décisions (DT : decision tree) de type régressif (il existe deux type de DT : ceux servant à des tâches de classification et ceux de

régression) d'où le terme de random forest étant donné que cette méthode utilise une multitude de ceux-ci. Dans le cadre de ce travail une population de 100 arbres ont été utilisés et la librairie RandomForestRegressor a été utilisée [Skl]. Celle-ci utilise une technique de regroupement de prédictions se basant sur une technique dite de boosting car elle ne va pas utiliser un système de vote comme le bagging mais une moyenne des prédictions pondérées selon leurs probabilités.

4. Résultats

Les résultats et tests présentés dans cette section ont été évalués selon des critères pratiques qui seront vus plus tard et des critères types de comparaison de prédiction à savoir le "root mean squared error" (RMSE), le "mean absolute error" (MAE), le "mean relative error" (MRE) et le "mean directionnal accuracy".
Ces critères ont été choisis car concernant les trois premiers ils se retrouvent comme critères de comparaison dans la majorité des travaux cités auparavant tel que [Div+18], [Div+19] ou encore [Li+17]. Concernant le dernier (MDA) celui-ci a été choisi car le fait de pouvoir prédire le plus efficacement possible la montée ou descente des valeurs est une caractéristique clé concernant la prédiction de manière efficace du dépassement d'un certain seuil. Les équations de ces différents coefficients sont les suivantes :

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} \hat{x_i} - x_i}{n}}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |\hat{x_i} - x_i|$$

$$MRE = \frac{1}{n}\sum_{i=1}^{n} \frac{|\hat{x_i} - x_i|}{x_i}$$

$$MDA = \frac{1}{n}\sum_{i=2}^{n} \mathbb{1}_{sign(x_i - x_{i-1}) == sign(\hat{x_i} - x_{i-1})}$$

Où x_i est la valeur au temps i, $\hat{x_i}$ est la prédiction de la valeur au temps i et $\mathbb{1}$ est la fonction indicatrice ou fonction caractéristique renvoyant 1 si l'élément appartient à l'ensemble et 0 dans le cas échéant (ici 1 si la direction prise par la time series est la bonne et inversement).
Concernant les tests destinés à la problématique de MEA il est intéressant de savoir si de manière générale lorsque la consommation dépasse un certain seuil est-ce que la prédiction l'a dépassée aussi et si oui est-ce en même temps ? Pour ce faire la fonction utilisée est celle présentée figure

4.1, celle-ci prenant en argument un threshold qui va influencer fortement les résultats obtenus donc pour se faire des valeurs ont été fixées arbitrairement. La raison pour laquelle ces valeurs ont été fixées au cas par cas est que certaines des zones de prédictions sont plus volatiles que d'autres ou ont des valeurs plus grandes (particulièrement la figure 3.4 montrait que les valeurs de ce cluster oscillaient entre 200 et 700 u). La philosophie suivie pour fixer ces valeurs a été de prendre une valeur dépassée de plus ou moins 10% de la valeur maximale et ce par le/les sommet(s) du graphique. Les seuils sont les suivants :

- Chiswick :
  - Sur une semaine avec 10 minutes d'intervalle : 500
  - Sur un mois avec 10 minutes d'intervalle : 500
- Lyndhurst :
  - Sur une semaine avec 1 minute d'intervalle : 200
- Chineham :
  - Sur une semaine avec 10 minutes d'intervalle : 200
  - Sur un mois avec 10 minutes d'intervalle : 200
  - Sur 3 mois avec 10 minutes d'intervalle : 175
  - Sur 6 mois avec 10 minutes d'intervalle : 175
  - Sur une semaine avec 1 minute d'intervalle : 150

Le but va donc être de minimiser les 3 premiers coefficients présentés et maximiser les autres. Sachant cela les résultats présentés ne seront que les meilleurs de leurs catégories et seront parfois mis en parallèle avec d'autres produits par la même technique pour faire ressortir de manière intrinsèque au modèle ses faiblesses et points forts.

```python
def goodAlertRate(val,pred,threshold):
    truePositive = 0
    falsePositive = 0
    trueNegative = 0
    falseNegative = 0


    for v in val:
        for p in pred:

            if(v <= threshold and p <=threshold):
                trueNegative+=1
            if(v>threshold and p>threshold):
                truePositive+=1
            if(v <= threshold and p>threshold):
                falsePositive+=1
            else:
                falseNegative+=1

    return truePositive, falsePositive, trueNegative, falseNegative
```

FIGURE 4.1 : La fonction servant à évaluer la capacité à identifier un dépassement d'un seuil fixé

4.1 Paramétrage des expériences

Voici pour chaque modèle une brève description des paramètres passés au méthodes créant ceux-ci et ce qu'ils veulent dire lorsque cela est pertinent pour l'expérience. Les paramètres non-explicités sont de manière générale des paramètres affichant des données durant la compilation pour aider au debuggage et non des paramètres déterminants.

4.1.1 ARMA et ARIMA

Pour rappel les données utilisées sont celles ayant été différenciées (la valeur en t a donc été remplacée par la soustraction de celle 144/1440 timestamps plus tard à elle-même). La fonction utilisée est la fonction auto-arima de la librairie Pyramids [Pyr] et ses paramètres utilisés sont repris sur la figure 4.2.

```
model = pm.auto_arima(training_data, start_p=0, start_q=0,
                      max_p=20, max_q=20, # maximum p and q
                      d=0,           # let model determine 'd' or not
                      seasonal=False,   # Seasonality
                      trace=True,
                      stepwise=True)
```

FIGURE 4.2 : La fonction utilisée pour le modèle ARMA/ARIMA

4.1.2 SARIMA

Les données de SARIMA contrairement à ARMA et ARIMA n'auront pas besoin d'être traitées car les paramètres (P, D, Q) correspondant aux paramètres de ARIMA mais dans ce cas-ci pour la saisonnalité s'en chargeront. A cela s'ajoute m qui est la taille du cycle à fournir à la fonction automatique. Les paramètres utilisés sont donc presque les mêmes que le modèle ARMA à l'exception que ici comme la figure 4.3 le montre la saisonnalité est inclue dans le paramétrage.

4.1.3 LSTM

Dans le cadre de LSTM la fonction utilisée vient de la librairie Keras [Ker] et comme le montre la figure 4.4 celle-ci nécessite quelques réglages. Voici l'explication dans l'ordre des opérations effectuées sur cette figure :

1. Le modèle est initialisé et stocké

```
smodel = pm.auto_arima(training_data, start_p=1, start_q=1,start_P=0,
                       max_p=3, max_q=3, m=12,
                       seasonal=True,
                       d=None, D=None,
                       trace=True,
                       stepwise=True)
```

FIGURE 4.3 : La fonction utilisée pour le modèle SARIMA

2. Le modèle est spécifié (LSTM) et la taille des matrices prises en argument est spécifiée par le `input_shape`
3. On spécifie le nombre d'entrées à gérer en même temps, ici une seule
4. On compile le modèle en lui demandant d'utiliser la méthode `mean_squared_error` pour calculer l'erreur durant l'entraînement et l'optimiseur `adam`
5. On entraîne nôtre modèle et on définit quelques valeurs concernant celui-ci

Le paramètre qui est surtout intéressant de discuter est l'optimiseur, celui-ci ayant été choisi car il est plus adapté aux grandes quantités de données et plus particulièrement celles non-linéaires [masa].

```
model = tf.keras.Sequential()
model.add(LSTM(4, input_shape=(1, previous)))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
history=model.fit(X_train, Y_train, validation_split=0.2, epochs=100, batch_size=448, verbose=2)
```

FIGURE 4.4 : La fonction utilisée pour le modèle LSTM

4.1.4 Random forest

La fonction utilisée pour créer le modèle RF nécessite elle très peu de paramétrage ce qui la rend très facile à utiliser. Celle-ci vient de la librairie Sklearn [Skl] et permet comme le montre la figure 4.5 de construire le modèle en deux lignes de code.

```
RF_Model = RandomForestRegressor(n_estimators=100, oob_score=True)
rgr=RF_Model.fit(features, labels)
```

FIGURE 4.5 : La fonction utilisée pour le modèle RF

Ici les paramètres utilisés permettent de définir le nombre d'arbres que l'on utilise dans le modèle à savoir 100 et le `oob_score` lui permet d'approximer la direction que devra prendre une valeur non-vue dans l'arbre (rappel une valeur entrant dans un arbre de décision va être triée jusqu'à atteindre une feuille).

4.2 ARMA et ARIMA

ARMA et ARIMA comme vu précédemment sont des modèles ne gérant pas la saisonnalité et devant donc utiliser des données ayant été modifiées pour supprimer celle-ci avant de pouvoir travailler avec. La fonction avec laquelle ces modèles ont été produits [Pyr] dans son exécution procède d'abord à un test permettant d'identifier si une tendance (pour rappel il s'agît du I de ARIMA) est présente dans les données et après avoir fait passer le test à tous les jeux de données cités auparavant il s'est avéré qu'aucun d'entre eux ne possédait de tendance ce qui signifie que le degré de différenciation d vaut 0 et le modèle est donc équivalent à un modèle ARMA possédant les mêmes p et q. Comme énoncé auparavant ces modèles n'ont pour unique but de permettre la comparaison car comme l'atteste l'état de l'art et particulièrement [Jur+15] qui pointe le fait qu'elles ne produisent pas de résultats particulièrement bons ni mauvais, ce qui se confirme sur la figure 4.6 représentant une des meilleures prédictions produites (en vert) par cette technique.

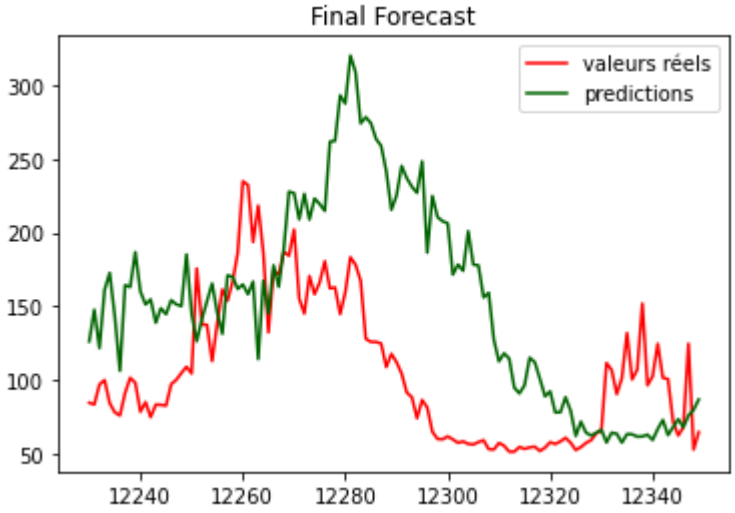

FIGURE 4.6 : La prédiction de 120 observations du modèle ARMA pour Chineham sur 3 mois avec 10 minutes d'espacement

Comme la figure 4.6 le montre la prédiction ne semble pas fortement précise mais ressemble malgré tout au comportement des valeurs réelles. Cette perte de précision est notamment due à la manière dont le problème de la saisonnalité a été réglé qui est une façon simplifiée. Pour rappel ces résultats serviront uniquement de point de comparaison afin de mesurer l'efficacité des autres méthodes.

4.3 SARIMA

Comme expliqué précédemment SARIMA contrairement à ARMA et ARIMA lui ne doit pas utiliser des données différenciées car la saisonnalité sera gérée par le modèle. Cependant durant la recherche un problème de puissance de calcul se pose dû au fait que m (ou la durée du cycle) est trop grand (de l'ordre de 24h ce qui représente 144 observations dans le cadre d'une série avec 10 minutes d'intervalle et 1440 dans celles de une). L'erreur semblerait liée à l'incapacité des outils utilisés (à savoir Jupyter notebook [Jup]) à gérer des calculs matriciels trop grands engendré par la fonction automatique de Pyramids [Pyr]. Pour pouvoir utiliser SARIMA il faut donc changer les données afin de réduire ce m à défaut de posséder une puissance de calcul suffisante. Avant donc d'appliquer la fonction calculant les différents(p, d, q) et (P, D, Q) optimaux il faut regrouper les données par groupe de 2 heures (ce qui représente 12 et 120 observations pour un espacement de 10 et 1 minutes) en calculer la moyenne et utiliser cette nouvelle série possède un m de 12 les deux cas au lieu de 144 et 1440. Les données étant changées pour ce cas particulier la période prédite le sera par conséquent elle aussi et correspondra toujours à 120 observations ce qui fera 10 jours au lieu de 20 heures et 2 heures. Cette nouvelle période de prédiction étant supérieure à une semaine les séries inférieures à cet intervalle de temps ne peuvent donc pas être étudiées avec SARIMA. D'autre part cette transformation va bien entendu fausser la comparaison mais les résultats n'en restent pas moins intéressants comme le montre la figure 4.7.

La figure 4.7 représente la meilleure prédiction faite avec SARIMA et a été réalisée sur la série la plus longue faisant 6 mois. Maintenant il est intéressant de regarder les coefficients énoncés précédemment afin de comparer les résultats produits par chaque série. Comme le montre la figure numéro 4.8 représentant l'ensemble des résultats obtenus la même série que celle de la figure 4.7 mais sur 3 mois a un RMSE plus faible ce qui signifie que les prédictions sont meilleures sur ce point-là mais la différence de MRE est très faible et le gain en MDA lui est par contre très grand ce qui fait que la série de 6 mois est préférable au vu de la problématique de MEA.

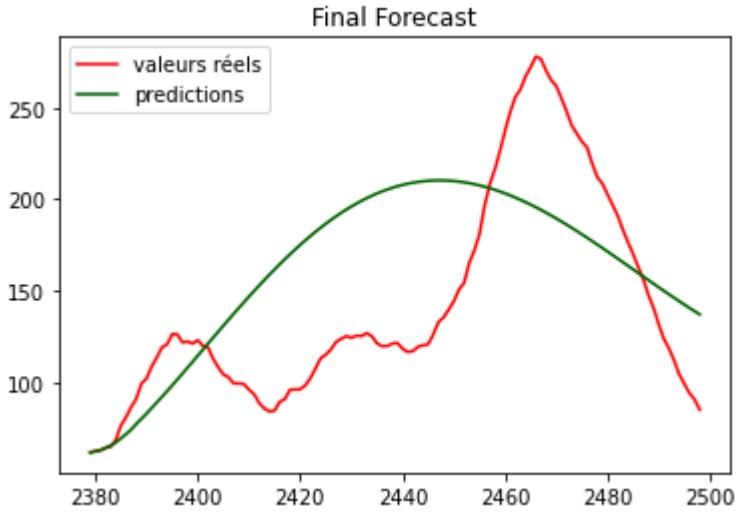

FIGURE 4.7 : La prédiction de 120 observations du modèle SARIMA pour Chineham sur 6 mois avec 2 heures d'espacement

	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
CC_10_min_mois	201,32	157,84	0,41	0,32	0,46	nan	0
CRG_10_min_mois	106,46	86,82	0,56	0,4	0,42	nan	0
CRG_10_min_3_mois	47,6	41,06	0,35	0,44	0,46	nan	0
CRG_10_min_6_mois	55,66	46,44	0,36	0,63	0,39	0,2	0,05

FIGURE 4.8 : L'ensemble des coefficient produit pas le modèle SARIMA

4.4 LSTM

Dans le cadre de LSTM afin d'obtenir les meilleurs résultats plusieurs paramètres ont été étudiés afin de trouver si éventuellement une combinaison de ceux-ci sortirait du lot. Les paramètres sont donc :

- La période étudiée : comment la quantité de donnée influence les résultats
- L'intervalle entre les observations : comme vu précédemment il y a des intervalles différents pour certaines séries
- La taille des sous-séries passées au modèle en entrée : cf la partie explication sur le fonctionnement de LSTM
- Le cluster étudié : tous n'agissent pas de la même façon et n'ont pas des valeurs similaires.

Les tailles testées des sous-séries ont été fixées à 12, 60, 120, 240, 360, 480 et 600 celles-ci étant comme soit $\frac{1}{10}$, $\frac{1}{2}$, 1, 2, 3, 4 et 5 fois la taille de la prédiction souhaitée. Les résultats calculés ont donc la forme présentée sur la figure 4.9 et seront diviser en deux catégories :

1. Les résultats sur des séries ayant pour intervalle de temps une minute
2. Les résultats sur des séries ayant pour intervalle de temps 10 minutes

Cette première distinction est faite car comme dans le premier cas les observations se suivent de plus près, l'écart entre deux valeurs consécutives est moins grand de manière générale (en une minute la consommation électrique a moins de temps pour varier que si elle en avait 10) ce qui rend par conséquent les coefficients étudiés moins grands.

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	87,27	73,16	1,03	0,44	0,26	0,1	0,15
60	93,98	75,4	1,06	0,4	0,35	0,1	0,06
120	38,11	32,1	0,38	0,61	0,43	0,1	0,01
240	33,25	26,14	0,27	0,55	0,47	nan	0
360	37,01	30,12	0,35	0,55	0,43	0,1	0,01
480	55,53	44,12	0,47	0,52	0,42	0,1	0,02
600	43,12	34,56	0,39	0,45	0,43	0,1	0,01

FIGURE 4.9 : Les résultats du modèle LSTM sur 3 mois de Chineham avec une espacement de 10 minutes

4.4.1 Avec un intervalle de une minute entre chaque observation

Dans cette partie le nombre de séries étudiées est de deux comme vu précédemment. Ces séries concernent le cluster de Lyndhurst et de Chineham et ce sur une durée de une semaine donnant donc des prédictions de 2 heures.
Les deux prédictions présentées sur la figure 4.10 seront seulement présentées visuellement dans cette section car le nombre de tableaux étant de 8 par modèle cela surchargerait la lecture de ce mémoire c'est pourquoi ces tableaux seront mis en annexe. Cependant même si les coefficients ne sont pas présentés ils seront discutés dans la partie conclusion.

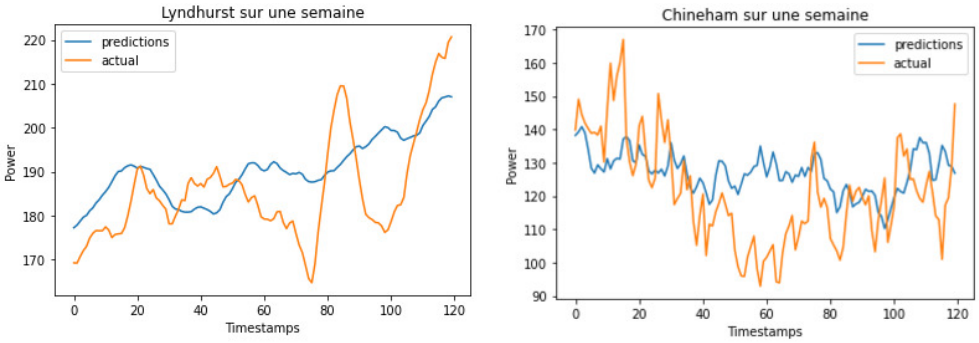

FIGURE 4.10 : Les résultats du modèle LSTM sur les séries de une semaine avec espacement de 1 minute de Lyndhurst et Chineham

Dans le cas de Lyndhurst la taille des sous-séries passées au modèle est de 480 (8 heures) tandis que pour Chineham il s'agît de 600 (ou 10 heures). Ces deux périodes étant respectivement la deuxième et la plus grande étudiée des tentatives avec des tailles encore plus grandes ont été menées et ont confirmé que les prédictions ne faisaient que se détériorer au fur et à mesure que cette taille augmentait.

4.4.2 Avec un intervalle de 10 minutes entre chaque observation

Dans cette partie le nombre de séries étudiées est de six et ces séries concernent le cluster de Chiswick et Chineham sur une durée allant de une semaine à un mois pour le premier et 6 pour le deuxième le tout donnant des prédictions de 20 heures.
Comme pour la partie précédente ici seul deux graphiques seront présentés sur la figure 4.11 représentant les meilleures prédictions de chaque cluster, Chiswick à gauche et Chineham à droite. Dans ce cas-ci la variante que représente la quantité de donnée disponible prend donc place étant donné

que le cluster Chiswick avec 10 minutes d'espacement contient 2 séries de tailles différentes et celui de Chineham 4. Les graphiques sont donc les résultats donnés par la meilleur combinaison des 4 paramètres cités plus haut.

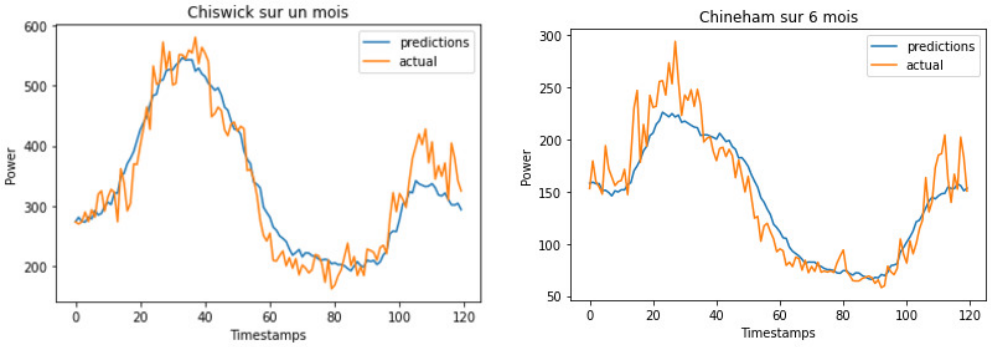

FIGURE 4.11 : Les résultats du modèle LSTM sur les séries de un mois (Chiswick) et 6 mois (Chineham) avec espacement de 10 minutes

Dans les deux cas il s'agît donc de l'intervalle le plus grand disponible du cluster ce qui est logique mais l'intérêt de l'utilisation de la quantité la plus grande disponible de donnée et si le gain de précision vaut la peine ou non de construire un modèle avec plus de données sera discuté plus tard. Les tailles choisies cette fois-ci sont 360 (soit 2 jours et 12 heures) pour Chiswick et 480 (ou 3 jours et 8 heures) pour Chineham. Il est important de souligner que Chiswick a un spectre de valeur allant de 200 à 700 tandis que Chineham lui possède des valeurs entre 50 et 350 ce qui rend les RMSE de Chineham meilleurs (ici il est de 22.32 contre 34.41 pour Chiswick) même si comme le graphique la prédiction de Chiswick semble être meilleure. C'est ici que la MRE prend tout sont sens car ici l'erreur relative moyenne de Chiswick est de 8% contre 10 pour Chineham d'où l'importance de ce coefficient.

4.5 Random forest

Le fonctionnement de la Random forest étant fort similaire à celui de LSTM au niveau de la forme des données passées au modèle la structure de cette partie sera donc la même.

4.5.1 Avec un intervalle de une minute entre chaque observation

Comme dans la partie LSTM cette première sous-section ne regarde que 2 séries appartenant à 2 clusters et les deux étant sur une durée de 1 semaine. Les meilleures prédictions pour les 2 séries étudiées ici comme le montre la figure 4.12 en comparant celles-ci à celles obtenues avec LSTM pour les mêmes séries il ressort que les prédictions s'apparentent plus à une droite représentant la valeur moyenne plutôt qu'à une prédiction fidèle du comportement de la série et ce pour une raison encore inconnue. Les paramètres utilisés étant les mêmes dans les deux cas le problème vient donc probablement de la série en elle-même et par conséquent celle-ci sera écartée pour ne pas fausser les résultats.

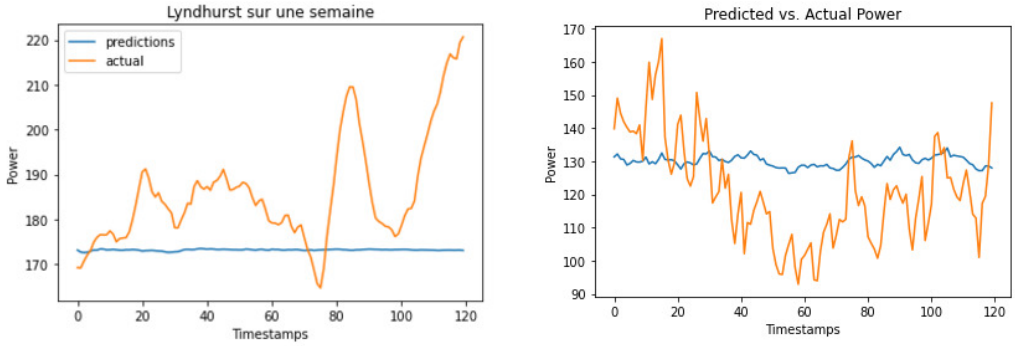

FIGURE 4.12 : Les résultats du modèle RF sur les séries de une semaine avec espacement de 1 minute de Lyndhurst et Chineham

Les tailles des sous-séries passées aux modèles sont toutes les deux 600 et donc comme pour LSTM des valeurs supérieures ont été testées et ont confirmé que les coefficients ne s'amélioraient pas ou alors en dépit d'une perte d'efficacité sur d'autres coefficients. En comparant ces résultats à ceux de LSTM on peut conclure que sur ces clusters-ci LSTM est plus performant car sa prédiction est plus proche de la réalité même si ses coefficients ne sont pas spécialement meilleurs. De plus dans cadre de RF les tailles des sous-séries sont plus grandes ce qui impacte la rapidité du modèle or celle-ci est une composante à tenir en compte lorsque l'on travaille à court terme.

4.5.2 Avec un intervalle de 10 minutes entre chaque observation

Toujours de même que pour LSTM ici 6 séries ont été comparées provenant des clusters Chiswick et Chineham sur des périodes allant de une semaine à 6 mois. Les séries encore une fois sont celles contenant le plus de données pour chacune d'entre elles et les prédictions ressemblent fortement à leurs correspondantes présentées précédemment avec une légère perte de précision qui dans les chiffres correspond à un RMSE plus grand de 1 pour Chiswick et de 5 pour Chineham ce qui est infime.

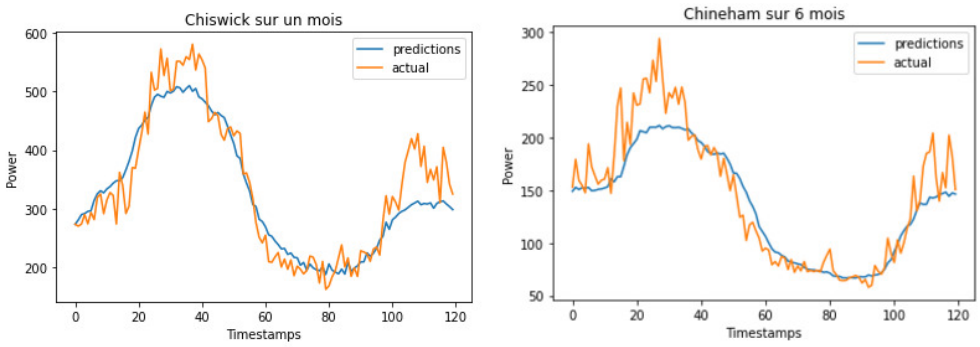

FIGURE 4.13 : Les résultats du modèle RF sur les séries de un mois (Chiswick) et 6 mois (Chineham) avec espacement de 10 minutes

Les tailles optimales dans ce cas-ci sont 360 pour Chiswick (ce qui est la même taille que dans la partie LSTM) et 600 pour Chineham (ce qui est une augmentation de 20 heures). Dans le cadre de l'entraînement des modèles le même appareil a été utilisé c'est pourquoi il est possible de comparer de manière générale la rapidité d'entraînement des deux techniques et il s'avère que le temps d'entraînement de RF augmente de manière considérable lorsque la taille des sous-suites augmente et cela combiné à des quantités de données assez grande entraîne une lenteur que l'on ne retrouve pas chez LSTM et qui est non négligeable dans le cadre du court-terme si la quantité de données devait augmenter.

4.6 Comparaison des coefficients

Dans cette section la seule différence qui sera faite sera pour les séries espacées en minutes et en dizaines de minutes (que SARIMA rejoindra même si les résultats sont différents dû au problème énoncé auparavant) afin de comparer les meilleures prédictions toute catégorie confondue : donc il n'y aura pas de différenciation fait quand au cluster concerné, la taille de la série entière et la taille des sous-séries mais uniquement l'espacement.

4.6.1 Avec un intervalle de une minute entre chaque observation

Le tableau reprenant tous les coefficients sur la figure 4.14 montre que LSTM et RF ont de loin les meilleures performances avec les meilleurs coefficients loin devant ARMA. Il est important de souligner aussi que les résultats de ARMA bien que ne semblant pas si mauvais que ça avec un RMSE et un MAE modérément plus grand que les meilleurs des deux autres sa MAE est quand à elle de quasiment 100%. Quand aux coefficients concernant la problématique de MEA ceux-ci sont sujet à interprétation de par leurs calcul assez éloigné de la réalité : une prédiction correspondant à une droite surplombant toutes les valeurs qui auraient due être prédite aura des coefficients assez bons même si dans la pratique cela aurait des résultats complètement faux c'est pourquoi la MDA ajoutée aux 3 coéfficients précédents est préférable afin de déterminer la qualité de la résolution du cas de MEA.

Modèle	Cluster	Période	Taille des inputs	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
ARMA(3,3)	Chineham	Une semaine	nan	30,77	25,48	3,11	0,49	nan	nan	0
SARIMA	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
LSTM	Lyndhurst	Une semaine	480	14,36	11,62	0,06	0,57	0,39	0,14	0,05
RF	Chineham	Une semaine	600	16,55	14,11	0,12	0,56	0,48	nan	0

FIGURE 4.14 : Les résultats tout confondu avec espacement de 1 minute

4.6.2 Avec un intervalle de 10 minutes entre chaque observation

Comme pour les résultats précédents les modèle LSTM et RF sont devant les autres sans pour autant se démarquer l'un de l'autre. Dans ce tableau-ci on voit aussi que SARIMA a des résultats plutôt satisfaisant avec une diminution de 75% de MRE comparé à ARMA ce qui confirme encore une fois le besoin de gérer la saisonnalité présente dans les données de manière efficace. La grande différence entre les performances de SARIMA et de ARMA est donc dû au fait que dans le premier la

saisonnalité a été entièrement prise en compte tandis que dans le deuxième elle est encore présente comme vu dans les graphes d'auto-corrélation.

Modèle	Cluster	Période	Tailles des inputs	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
ARMA(2,1)	Chineham	3 mois	nan	25,81	18,94	1,1	0,5	0,48	nan	0
SARIMA(2, 0, 2)x(0, 0, 2, 12)	Chineham	3 mois	nan	47,6	41,06	0,35	0,44	0,46	nan	0
LSTM	Chineham	6 mois	480	22,32	15,96	0,1	0,57	0,36	0,34	0,1
RF	Chineham	6 mois	600	22,5	15,78	0,09	0,57	0,35	0,34	0,1

FIGURE 4.15 : Les résultats tout confondu avec espacement de 10 minutes

5. Conclusion et perspectives

Pour rappel le mémoire porte sur le sujet de la prédiction de la consommation électrique d'une population donnée avec une attention particulière pour la capacité à identifier le mouvement que prend la série temporelle afin que la prédiction puisse annoncer le dépassement d'un seuil le plus efficacement possible. Pour ce faire un maximum de techniques ont été étudiées pour pouvoir prédire au mieux la demande future en électricité et ainsi permettre aux fournisseurs d'ajuster au mieux leurs distributions afin d'éviter une surcharge du réseau électrique.
Parmi les méthodes étudiées, deux sont sorties du lot sans pour autant se démarquer l'une de l'autre en terme de précision cependant lors de l'entraînement des modèles il s'est avéré que LSTM surpasse RF largement en terme de vitesse. RF a du mal à gérer les plus grandes quantités de données ce qui est un point faible majeur étant donné que les performances augmentent en parallèle avec la quantité de données. Un autre résultat intéressant qui a été souligné par les résultats présentés est que la taille optimale des sous-séries passées aux modèles se situerait aux alentour des 480 observations ce qui semble logique en reprenant le graphe d'auto-corrélation de la figure 3.10 où les lags ne semblent plus avoir d'impact sur la valeur à prédire à partir du 470$^{\text{ième}}$. Un dernier résultat intéressant est que peu importe le cluster étudié les résultats augmentent proportionnellement à la quantité de données disponibles comme l'illustre la figure 5.1 montrant l'évolution du RMSE en fonction de la quantité de donnée disponible. Ce graphique semble aussi montrer qu'il y a un premier optimum local aux alentours de un mois pouvant éventuellement donner à la firme des résultats satisfaisants sans devoir utiliser une trop grande quantité de données.
Concernant les futurs travaux possibles sur le sujet il faudrait avant tout pouvoir disposer d'informations plus complètes. Les données présentes dans le data-set étant terriblement documentées leurs utilités est laissée à l'imagination du lecteur et ne peuvent faire l'objet d'un travail scientifique consciencieux. Cependant comme il l'a été énoncé dans l'introduction et l'état de l'art il serait intéressant de posséder des données plus complètes sur les véhicules électriques, tel que leurs impacts sur la consommation totale afin de mettre celle-ci en parallèle avec la consommation individuelle

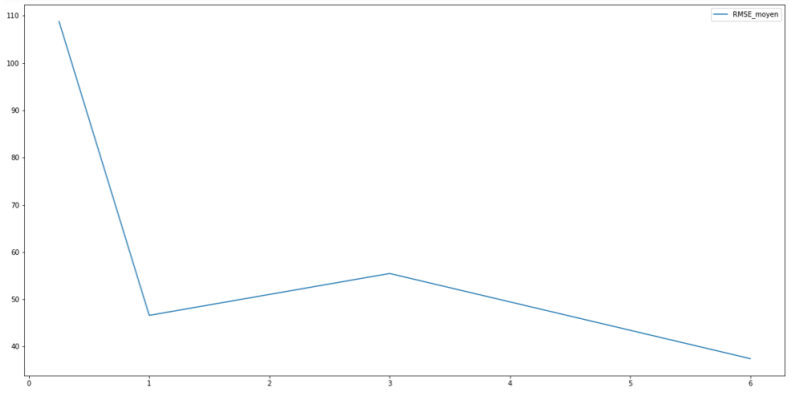

FIGURE 5.1 : L'efficacité marginale de LSTM en fonction de la période disponible en mois

d'un cluster et voir dans quelles mesures la limitation du temps de charge serait efficace. Par la suite une étude de l'impacte de cette limitation de temps de charge sur l'utilisation de son propriétaire avec les données sur ces trajets pourrait être réalisée afin de voir si la solution ne gênerait pas trop les ménages possédant un véhicule électrique. D'autre part ces travaux pourrait être mis en relation avec celui cité dans l'état de l'art [Zuf+12] afin d'identifier les véhicules électriques à l'échelle d'un cluster et non plus d'un ménage.

Bibliographie

[War43] Walter Pitts Warren S. McCulloch. "A logical calculus of the ideas immanent in nervous activity". In : (1943). DOI : https://doi.org/10.1007/BF02478259.

[Whi51] P. Whittle. *Hypothesis Testing in Time Series Analysis*. Statistics / Uppsala universitet. Almqvist & Wiksells boktr., 1951. URL : https://books.google.es/books?id=tteaMQEACAAJ.

[Sch78] Gideon Schwarz. "Estimating the Dimension of a Model". In : *Ann. Statist.* 6.2 (mar. 1978), p. 461-464. DOI : 10.1214/aos/1176344136. URL : https://doi.org/10.1214/aos/1176344136.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton et Ronald J. Williams. "Learning representations by back-propagating errors". In : 323.6088 (oct. 1986), p. 533-536. DOI : 10.1038/323533a0.

[Tin95] Tin Kam Ho. "Random decision forests". In : *Proceedings of 3rd International Conference on Document Analysis and Recognition*. T. 1. 1995, 278-282 vol.1.

[WR96] M. Wilson et P. Rolfe. "The path integral formalism applied to photon migration in turbid media". In : *Proceedings of 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. T. 2. 1996, 844-845 vol.2.

[KK01] J. F. Kolen et S. C. Kremer. "Gradient Flow in Recurrent Nets : The Difficulty of Learning LongTerm Dependencies". In : *A Field Guide to Dynamical Recurrent Networks*. 2001, p. 237-243.

[POP08] Luis Pérez-Lombard, José Ortiz et Christine Pout. "A review on buildings energy consumption information". In : *Energy and Buildings* 40.3 (2008), p. 394-398. ISSN : 0378-7788. DOI : https://doi.org/10.1016/j.enbuild.2007.03.007. URL : http://www.sciencedirect.com/science/article/pii/S0378778807001016.

[Far10] H. Farhangi. "The path of the smart grid". In : *IEEE Power and Energy Magazine* 8.1 (2010), p. 18-28.

[Zuf+12] D. Zufferey et al. "Machine learning approaches for electric appliance classification". In : *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*. 2012, p. 740-745.

[KMM14] Nasrin Khansari, Ali Mostashari et Mo Mansouri. "Conceptual Modeling of the Impact of Smart Cities on Household Energy Consumption". In : t. 28. Déc. 2014, p. 81-86. doi : 10.1016/j.procs.2014.03.011.

[Jur+15] Sergio Jurado et al. "Hybrid methodologies for electricity load forecasting : Entropy-based feature selection with machine learning and soft computing techniques". In : *Energy* 86 (2015), p. 276-291. issn : 0360-5442. doi : https://doi.org/10.1016/j.energy.2015.04.039. url : http://www.sciencedirect.com/science/article/pii/S0360544215004934.

[BRF16] M.A. Rafe Biswas, Melvin D. Robinson et Nelson Fumo. "Prediction of residential building energy consumption : A neural network approach". In : *Energy* 117 (2016), p. 84-92. issn : 0360-5442. doi : https://doi.org/10.1016/j.energy.2016.10.066. url : http://www.sciencedirect.com/science/article/pii/S0360544216315006.

[BR17] Riccardo Bonetto et Michele Rossi. "Machine Learning Approaches to Energy Consumption Forecasting in Households". In : *CoRR* abs/1706.09648 (2017). arXiv : 1706.09648. url : http://arxiv.org/abs/1706.09648.

[Gre+17] K. Greff et al. "LSTM : A Search Space Odyssey". In : *IEEE Transactions on Neural Networks and Learning Systems* 28.10 (2017), p. 2222-2232.

[Li+17] Chengdong Li et al. "Building Energy Consumption Prediction : An Extreme Deep Learning Approach". In : *Energies* 10.10 (oct. 2017), p. 1525. issn : 1996-1073. doi : 10.3390/en10101525. url : http://dx.doi.org/10.3390/en10101525.

[ZK17] Fikirte Zemene et Vijayshri Khedkar. "Survey on Machine Learning based Electric Consumption Forecasting using Smart Meter Data". In : *International Journal of Computer Applications* 180 (2017), p. 46-52.

[Div+18] Federico Divina et al. "Stacking Ensemble Learning for Short-Term Electricity Consumption Forecasting". In : *Energies* 11.4 (avr. 2018), p. 949. issn : 1996-1073. doi : 10.3390/en11040949. url : http://dx.doi.org/10.3390/en11040949.

[Wei+18] Yixuan Wei et al. “A review of data-driven approaches for prediction and classification of building energy consumption”. In : *Renewable and Sustainable Energy Reviews* 82 (2018), p. 1027-1047. ISSN : 1364-0321. DOI : https://doi.org/10.1016/j.rser.2017.09.108. URL : http://www.sciencedirect.com/science/article/pii/S136403211731362X.

[Div+19] Federico Divina et al. “A Comparative Study of Time Series Forecasting Methods for Short Term Electric Energy Consumption Prediction in Smart Buildings”. In : *Energies* 12.10 (mai 2019), p. 1934. ISSN : 1996-1073. DOI : 10.3390/en12101934. URL : http://dx.doi.org/10.3390/en12101934.

[PB19] Aaditi Parate et Sachin Bhoite. “Individual Household Electric Power Consumption Forecasting using Machine Learning Algorithms-**********”. In : *International Journal of Computer Applications Technology and Research* 8 (sept. 2019). DOI : 10.7753/IJCATR0809.1007.

[Ave] My Electric Avenue. *My Electric Avenue official website*. URL : http://myelectricavenue.info/. (accessed : 05.03.2020).

[Bro] Jason Brownlee. *How to Decompose Time Series Data into Trend and Seasonality*. URL : https://machinelearningmastery.com/decompose-time-series-data-trend-seasonality/. (accessed : 9.04.2020).

[Jup] Jupyter. *Official website*. URL : https://jupyter.org/. (accessed : 03.04.2020).

[Ker] Keras. *Keras website*. URL : https://keras.io/. (accessed : 16.04.2020).

[masa] Machine learning mastery. *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. URL : https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/. (accessed : 15.04.2020).

[masb] Machine learning mastery. *How to Create an ARIMA Model for Time Series Forecasting in Python*. URL : https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/. (accessed : 20.05.2020).

[Max] Maxicours. *Puissance en triphasé*. URL : https://www.maxicours.com/se/cours/puissance-en-triphase/. (accessed : 1.04.2020).

[plu] Machine learning plus. *ARIMA Model – Complete Guide to Time Series Forecasting in Python*. URL : https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/. (accessed : 10.04.2020).

[Pyr] PYRAMID. *pyramid.arima.auto$_a$rimalibrary*. URL : http://alkaline-ml.com/pmdarima/0.9.0/index.html. (accessed : 14.04.2020).

[Rit] RITVIKMATH. *Time Series Talk : Stationarity*. URL : hhttps://www.youtube.com/watch?v=oY-j2Wof51c. (accessed : 9.04.2020).

[Skl] SKLEARN. *sklearn.ensemble.RandomForestRegressor*. URL : https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html. (accessed : 30.04.2020).

[Smi] C. SMITH. *Un exemple de réseau de neuronne*. URL : https://fx-mon.blogspot.com/2011/12/meta-trader-sunqu-review.html. (accessed : 20.05.2020).

[Sta] STACKOVERFLOW. *Structure d'un neurone LSTM*. URL : https://i.stack.imgur.com/RHNrZ.jpg. (accessed : 25.04.2020).

[Wik] WIKIPEDIA. *La structure d'un LSTM*. URL : https://en.wikipedia.org/wiki/Long_short-term_memory. (accessed : 20.05.2020).

6. Annexes

6.1 Résultats LSTM

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	13,61	9,9	0,05	0,54	0,37	0,14	0,06
60	25,29	20,46	0,11	0,5	0,29	0,14	0,15
120	25,4	22,5	0,12	0,53	0,19	0,14	0,3
240	25,68	23,68	0,13	0,6	0,18	0,14	0,34
360	26,48	23,95	0,13	0,63	0,19	0,14	0,32
480	14,36	11,62	0,06	0,57	0,39	0,14	0,05
600	24,62	22,11	0,12	0,61	0,19	0,14	0,32
720	21,68	17,82	0,09	0,65	0,3	0,14	0,14
840	67,09	58,77	0,3	0,44	0,46	nan	0

FIGURE 6.1 : BL sur une semaine avec 1 minute d'espacement

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	143,35	120,8	0,42	0,48	0,45	nan	0
60	117,32	97,11	0,34	0,52	0,45	nan	0
120	63,68	55,1	0,19	0,5	0,42	0,15	0,02
240	44,05	37,82	0,12	0,5	0,41	0,15	0,03
360	45,52	38,2	0,11	0,52	0,43	0,15	0,02
480	42,57	35,94	0,11	0,48	0,42	0,15	0,02
600	43,08	33,72	0,09	0,41	0,42	0,15	0,02

FIGURE 6.2 : CC sur une semaine avec 10 minutes d'espacement

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	123,91	102,12	0,32	0,49	0,45	nan	0
60	194,52	148,07	0,59	0,52	0,4	0,15	0,04
120	44,78	36,62	0,13	0,52	0,41	0,15	0,03
240	40,78	33,06	0,11	0,56	0,43	0,15	0,01
360	34,41	27,6	0,08	0,58	0,42	0,15	0,02
480	36,57	28,41	0,08	0,61	0,42	0,15	0,02
600	38,31	30,8	0,09	0,58	0,42	0,15	0,02

FIGURE 6.3 : CC sur un mois avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	20,56	16,79	0,13	0,47	0,48	nan	0
60	15,47	12,77	0,1	0,46	0,48	nan	0
120	26,1	20,33	0,16	0,52	0,48	nan	0
240	53,42	44,44	0,38	0,53	0,34	0,04	0,03
360	109,88	95,35	0,83	0,51	0,12	0,04	0,2
480	44,56	37,91	0,33	0,47	0,23	0,14	0,23
600	15,13	12,01	0,1	0,48	0,48	nan	0
720	20,1	16,35	0,13	0,48	0,46	nan	0
840	24,45	19,02	0,15	0,53	0,45	0,14	0

FIGURE 6.4 : CRG sur une semaine avec 1 minute d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	69,33	57,8	0,55	0,59	0,43	nan	0
60	101,05	82,5	0,87	0,38	0,37	0,21	0,09
120	115,33	99,56	0,93	0,4	0,38	0,21	0,06
240	119,54	101,72	0,93	0,49	0,39	0,21	0,06
360	114,01	97,94	0,87	0,4	0,39	0,21	0,05
480	125,42	106,9	0,95	0,37	0,39	0,21	0,06
600	116,75	98,37	0,87	0,45	0,39	0,21	0,05

FIGURE 6.5 : CRG sur une semaine avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	75,95	64,85	0,58	0,42	0,43	nan	0
60	85,73	68,58	0,71	0,58	0,35	0,22	0,12
120	31,44	22,94	0,17	0,63	0,39	0,22	0,05
240	33,71	25,56	0,2	0,54	0,4	0,22	0,04
360	35,97	27,77	0,21	0,54	0,39	0,22	0,05
480	32,74	22,17	0,16	0,57	0,4	0,22	0,03
600	30,67	20,63	0,15	0,54	0,4	0,22	0,04

FIGURE 6.6 : CRG sur un mois avec 10 minutes d'espacement

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	87,27	73,16	1,03	0,44	0,26	0,1	0,15
60	93,98	75,4	1,06	0,4	0,35	0,1	0,06
120	38,11	32,1	0,38	0,61	0,43	0,1	0,01
240	33,25	26,14	0,27	0,55	0,47	nan	0
360	37,01	30,12	0,35	0,55	0,43	0,1	0,01
480	55,53	44,12	0,47	0,52	0,42	0,1	0,02
600	43,12	34,56	0,39	0,45	0,43	0,1	0,01

FIGURE 6.7 : CRG sur 3 mois avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	58,9	50,63	0,42	0,5	0,39	nan	0
60	63,01	54,34	0,52	0,48	0,39	0,34	0
120	23,1	17,36	0,13	0,57	0,36	0,34	0,1
240	33,54	28,04	0,22	0,53	0,34	0,34	0,14
360	30,88	23,88	0,17	0,57	0,36	0,34	0,08
480	22,32	15,96	0,1	0,57	0,36	0,34	0,1
600	30,37	25,56	0,17	0,58	0,37	0,34	0,07

FIGURE 6.8 : CRG sur 6 mois avec 10 minutes d'espacement

6.2 Résultats RF

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	32,71	29,83	0,12	0,15	0,44	nan	0
60	38,59	35,58	0,18	0,39	0,46	nan	0
120	30,17	25,63	0,13	0,5	0,46	nan	0
240	18,55	14,17	0,07	0,49	0,46	nan	0
360	34,72	29,55	0,15	0,42	0,46	nan	0
480	25,36	21,38	0,11	0,49	0,46	nan	0
600	17,5	13,34	0,06	0,51	0,46	nan	0
720	17,61	13,51	0,06	0,42	0,46	nan	0
840	17,68	13,49	0,06	0,54	0,46	nan	0

FIGURE 6.9 : BL sur une semaine avec 1 minute d'espacement

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	133,89	110,7	0,4	0,37	0,45	nan	0
60	127,11	102,9	0,37	0,45	0,45	nan	0
120	45,41	36,2	0,1	0,61	0,41	0,15	0,03
240	46,37	36,13	0,1	0,54	0,42	0,15	0,02
360	44,15	34,59	0,09	0,5	0,42	0,15	0,02
480	40,21	31,38	0,09	0,51	0,42	0,15	0,02
600	43,68	32,78	0,09	0,47	0,43	0,15	0,01

FIGURE 6.10 : CC sur une semaine avec 10 minutes d'espacement

Nombre d'input	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	129,31	102,88	0,29	0,47	0,45	nan	0
60	126,99	103,84	0,4	0,47	0,45	nan	0
120	37,86	28,87	0,08	0,54	0,43	0,15	0,02
240	38,78	29,26	0,08	0,6	0,44	0,15	0,01
360	35,04	26,55	0,08	0,69	0,43	0,15	0,01
480	37,81	27,52	0,08	0,63	0,43	0,15	0,02
600	37,67	28,7	0,08	0,63	0,42	0,15	0,02

FIGURE 6.11 : CC sur un mois avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	16,44	13,53	0,11	0,37	0,48	nan	0
60	16,68	13,32	0,1	0,49	0,48	nan	0
120	16,74	14,19	0,12	0,52	0,48	nan	0
240	17,67	15,28	0,13	0,4	0,48	nan	0
360	18,05	15,62	0,13	0,51	0,48	nan	0
480	17,59	15,17	0,13	0,46	0,48	nan	0
600	16,55	14,11	0,12	0,56	0,48	nan	0
720	17,42	14,97	0,13	0,42	0,48	nan	0
840	17,12	14,57	0,12	0,48	0,48	nan	0

FIGURE 6.12 : CRG sur une semaine avec 1 minute d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	66,56	55,28	0,51	0,58	0,43	nan	0
60	90,44	74,43	0,68	0,58	0,43	nan	0
120	103,69	88,97	0,8	0,4	0,4	0,21	0,03
240	109,17	92,45	0,83	0,47	0,4	0,21	0,04
360	88,18	76,34	0,6	0,52	0,43	nan	0
480	115,44	97,97	0,88	0,44	0,39	0,21	0,05
600	117,143	99,74	0,9	0,47	0,39	0,21	0,05

FIGURE 6.13 : CRG sur une semaine avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	74,46	58,99	0,45	0,53	0,43	nan	0
60	61,73	49,5	0,38	0,52	0,39	0,22	0,05
120	29,17	19,74	0,14	0,53	0,4	0,22	0,04
240	35,9	22,51	0,14	0,57	0,4	0,22	0,04
360	37,19	23,54	0,15	0,56	0,4	0,22	0,04
480	35,05	22,69	0,15	0,6	0,4	0,22	0,04
600	31,16	20,68	0,14	0,51	0,4	0,22	0,04

FIGURE 6.14 : CRG sur un mois avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	66,96	58,57	0,8	0,52	0,47	nan	0
60	41,23	32,54	0,36	0,58	0,4	0,1	0,03
120	40,08	31,27	0,32	0,52	0,4	0,1	0,03
240	35,2	27,03	0,28	0,61	0,44	0,1	0,01
360	32,92	25,11	0,26	0,48	0,43	0,1	0,01
480	36,14	27,49	0,29	0,59	0,43	0,1	0,01
600	37,24	28,88	0,31	0,55	0,43	0,1	0,01

FIGURE 6.15 : CRG sur 3 mois avec 10 minutes d'espacement

Nombre d'in	RMSE	MAE	MRE	MDA	Accuracy	Precision	Recall
12	53,03	45,5	0,36	0,53	0,39	nan	0
60	29,5	20,77	0,13	0,53	0,36	0,34	0,09
120	24,48	16,7	0,1	0,63	0,36	0,34	0,09
240	24,45	17,95	0,11	0,57	0,36	0,34	0,1
360	24,26	17,13	0,1	0,53	0,36	0,34	0,1
480	24,3	16,99	0,1	0,55	0,36	0,34	0,1
600	22,5	15,78	0,09	0,57	0,35	0,34	0,1

FIGURE 6.16 : CRG sur 6 mois avec 10 minutes d'espacement

7. Glossaire

SC	Smart city
SG	Smart grid
SM	Smart meter
ML	Machine learning
ANN	Artificial Neural Network
RF	Random Forest
FIR	Fuzzy Inductive Reasoning
ARIMA	AutoRegressive Integrated Moving Average
SVM	Support Vector Machine
NAR	Non AutoRegressive (Neural Network)
LSTM	Long-Short Term Memory (Neural Network)
ARMA	AutoRegressive Moving Average
ADAGRAD	Adaptive Gradient (AdaGrad)
SAE	Stack Auto-Encoder
ELM	Extreme Learning Machine
GBM	Generalized Boosted Model
PACF	Partial Auto-Correlation Function
ACF	Auto-Correlation Function
RNN	Recurrent Neural Network
DT	Decision tree
RMSE	Root Mean Squared Error
MAE	Mean Absolute Error
MRE	Mean Relative Error
MDA	Mean Directional Accuracy